Guide de l’utilisateur

AWS Gestion de l’identité et des accès



Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Gestion de l’identité et des accès: Guide de l’utilisateur

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Les marques et la présentation commerciale d’Amazon ne peuvent pas être utilisées en relation avec un produit ou un service qui n’appartient pas à Amazon, de toute manière susceptible de créer une confusion chez les clients ou de toute manière dénigrant ou discréditant Amazon. Toutes les autres marques commerciales qui ne sont pas la propriété d'Amazon sont la propriété de leurs propriétaires respectifs, qui peuvent ou non être affiliés ou connectés à Amazon, ou sponsorisés par Amazon.

Table of Contents

En quoi consiste IAM ? 1
Pourquoi utiliser IAM ? 3
Accès partagé à votre Compte AWS 3
Autorisations granulaires 3
Accès sécurisé à AWS ressources pour les applications qui s'exécutent sur Amazon EC2 4
Multi-factor authentification (MFA) 4
Fédération des identités 4
Informations d'identité par sécurité 4
Conformité PCI DSS 4
Quand utiliser IAM ? 5
Lorsque vous effectuez différentes activités professionnelles 5
Quand vous êtes autorisé à accéder AWS resources 6
Lorsque vous vous connectez en tant qu'utilisateur IAM 6
Lorsque vous assumez un rôle IAM 7
Lorsque vous créez des politiques et des autorisations 7
Comment gérer IAM ? 8
Utilisez le Console de gestion AWS 9
AWS Outils de ligne de commande 10
Utilisez le AWS SDKs 13
Utilisation de l'API Query IAM 13
Fonctionnement de IAM 13
Composants d'une requête 15
Comment les principaux sont authentifiés 15
Principes de base des politiques d'autorisation 16
..... 17
Comparaison des identités et des informations d'identification IAM 18
Termes 19
Différence entre les utilisateurs IAM et les utilisateurs dans IAM Identity Center 21
Fédération des utilisateurs à partir d'une source d'identité existante 22
Différentes méthodes pour fournir un accès aux utilisateurs 23
Prise en charge de l'accès programmatique des utilisateurs 27
Comment les autorisations et les politiques assurent la gestion des accès 29
Politiques et comptes 29
Politiques et utilisateurs 30

Politiques et groupes IAM 30
sessions d’utilisateur fédéré et rôle 31
Politiques basées sur les ressources et politiques basées sur l'identité 31
Définition des autorisations avec autorisation ABAC 33
Comparaison d’ABAC et du modèle RBAC traditionnel 33
Prise en main 36
Configuration de votre Compte AWS 37
Visualisation de votre Compte AWS ID 39
Pour consulter votre Compte AWS ID 39
Utiliser un alias pour votre Compte AWS ID 41
Création d’un alias de compte 42
Suppression d'un alias de compte 43
Planifiez l'accès à votre AWS compte 44
Sources d’identité 44
Gestion des accès 45
Cas d’utilisation pour les utilisateurs IAM 45
Utiliser l’authentification multifactorielle avec vos identités 60
Préparation des autorisations de moindre privilège 61
Révision des dernières informations consultées pour votre AWS compte 62
Génération d’une politique basée sur l’activité d’accès 68
Utilisation de la recherche pour trouver des ressources IAM 72
Bonnes pratiques de sécurité et cas d'utilisation 76
Bonnes pratiques de sécurité 76
Obliger les utilisateurs humains à utiliser la fédération avec un fournisseur d'identité pour accéder AWS en utilisant des informations d'identification temporaires 77
Exiger que les charges de travail utilisent des informations d'identification temporaires avec des rôles IAM pour accéder AWS 78
Authentification multifactorielle (MFA) nécessaire. 79
Mettre à jour les clés d'accès lorsque cela est nécessaire pour les cas d'utilisation nécessitant des informations d'identification à long terme 80
Suivez les meilleures pratiques pour protéger vos informations d'identification de l'utilisateur root 81
Accorder les autorisations de moindre privilège 81
Commencez avec AWS politiques gérées et évolution vers les autorisations de moindre privilège 82

Utiliser IAM Access Analyzer pour générer des politiques de moindre privilège en fonction de l'activité d'accès ... 82
Vérifiez et supprimez régulièrement les utilisateurs, les rôles, les autorisations, les politiques et les informations d'identification non utilisés ... 82
Utiliser des conditions dans les stratégies IAM pour restreindre davantage l'accès ... 83
Vérifiez l'accès public et intercompte aux ressources avec IAM Access Analyzer. ... 83
Utilisez IAM Access Analyzer pour valider vos politiques IAM pour des autorisations sécurisées et fonctionnelles ... 83
Établir des garde-fous d'autorisations sur plusieurs comptes ... 84
Utiliser les limites des autorisations pour déléguer la gestion des autorisations au sein d'un compte. ... 84
Bonnes pratiques d'utilisateur root ... 85
Sécurisez vos informations d’identification d'utilisateur root pour empêcher toute utilisation non autorisée ... 86
Utiliser un mot de passe utilisateur root fort pour protéger l'accès ... 87
Sécurisez votre connexion d'utilisateur root avec l'authentification multifactorielle (MFA) ... 87
Ne créer aucune clé d'accès pour l'utilisateur root ... 88
Utiliser une approbation par plusieurs personnes pour la connexion de l'utilisateur root dans la mesure du possible ... 88
Utiliser une adresse e-mail de groupe pour les informations d'identification de l'utilisateur root ... 89
Restreindre l'accès aux mécanismes de récupération des comptes ... 89
Sécurisez votre AWS Organizations informations d'identification de l'utilisateur root du compte ... 90
Surveiller l'accès et l'utilisation ... 91
Cas d'utilisation métier ... 93
Configuration initiale d'Example Corp ... 93
Cas d'utilisation pour IAM avec Amazon EC2 ... 95
Cas d'utilisation pour IAM avec Amazon S3 ... 96
Tutoriels ... 99
Déléguez l'accès à Comptes AWS tous les rôles ... 99
Considérations ... 101
Conditions préalables ... 102
Création d’un rôle dans le compte Destination ... 102
Accorder l'accès au rôle ... 106
Tester l'accès en changeant de rôles ... 109

Ressources supplémentaires 115
Résumé 115
Créer une politique gérée par le client 115
Conditions préalables 116
Étape 1 : Créer la politique 116
Étape 2 : Attacher la politique 117
Étape 3 : Tester l'accès utilisateur 118
Ressources connexes 118
Résumé 119
Utiliser le contrôle d'accès basé sur les attributs (ABAC, attribute-based access control) 119
Aperçu du didacticiel 120
Conditions préalables 122
Étape 1 : Créer des utilisateurs test 122
Étape 2 : Créer la politique ABAC 124
Étape 3 : Créer les rôles 129
Étape 4 : Tester la création de secrets 130
Étape 5 : Tester l'affichage des secrets 133
Étape 6 : Tester l'adaptabilité 135
Étape 7 : Tester la mise à jour et la suppression des secrets 137
Résumé 139
Ressources connexes 139
Utilisation de balises de session SAML pour le contrôle ABAC 140
Autoriser les utilisateurs à gérer leurs informations d'identification et leurs paramètres MFA 145
Conditions préalables 146
Étape 1 : Créer une politique pour appliquer l'authentification MFA 147
Étape 2 : Attacher des politiques à votre groupe d'utilisateurs test 148
Étape 3 : Test de l'accès utilisateur 148
Ressources connexes 151
Créez un IdP SAML avec CloudFormation 151
Conditions préalables 151
Créer l'IdP 152
Vérifier l'IdP 155
Supprimer des ressources 155
Détails du modèle 156
Modèle 156
Créez un rôle fédéré SAML avec CloudFormation 160

Conditions préalables 161
Créer le rôle fédéré 161
Tester le rôle fédéré 164
Supprimer des ressources 164
Détails du modèle 165
Modèle 165
Créez un IdP SAML et un rôle fédéré avec CloudFormation 171
Conditions préalables 172
Créer les ressources 172
Tester la fédération 176
Supprimer des ressources 176
Détails du modèle 177
Modèle 178
Identités 187
Utilisateur racine IAM 188
Utilisateurs IAM 188
Groupes d'utilisateurs IAM 188
Rôles IAM 188
Utilisateur racine d'un compte AWS 188
Gestion centralisée de l'accès root pour les comptes membres 189
Centralisation de l'accès racine 190
Exécuter une tâche privilégiée 195
MFA pour l'utilisateur racine 202
Modifier le mot de passe 211
Réinitialisation d'un mot de passe d'utilisateur racine perdu ou oublié 213
Création de clés d'accès pour l'utilisateur racine 215
Suppression d'une clé d'accès de l'utilisateur racine 218
Tâches nécessitant les informations d'identification de l'utilisateur root 219
Ressources supplémentaires 222
Utilisateurs 223
Comment ? AWS identifie un utilisateur IAM 223
Utilisateurs IAM et informations d'identification 224
Rôles et autorisations IAM 225
Utilisateurs et comptes IAM 226
Utilisateurs IAM en tant que comptes de service 226
Comment les utilisateurs d'IAM se connectent à AWS 227

Créez un utilisateur 232
Affichage des utilisateurs IAM 234
Renommer un utilisateur 237
Supprimer un utilisateur 239
Contrôle de l’accès utilisateur à la console 243
Modification des autorisations des utilisateurs 245
Mots de passe utilisateur 254
Gérer des clés d’accès 276
Multi-factor authentication 307
Informations d’identification spécifiques au service 392
Recherche d’informations d’identification non utilisées 403
Génération de rapports d’informations d’identification 408
Informations d'identification IAM pour CodeCommit 416
Gestion des certificats de serveur 419
Groupes d’utilisateurs 425
Création de groupes IAM 427
Afficher les groupes IAM 429
Modification des utilisateurs dans les groupes IAM 431
Attachement d’une politique à un groupe d’utilisateurs 432
Affectation d’un nouveau nom à un groupe d’utilisateurs 433
Supprimer un groupe IAM 435
Roles 437
Quand créer un utilisateur IAM (au lieu d'un rôle) 439
Termes et concepts relatifs aux rôles 440
Ressources supplémentaires 443
Le problème de l'adjoint confus 445
Scénarios courants 455
Création d’un rôle 470
Gestion des rôles 539
Méthodes pour assumer un rôle 578
Fournisseurs d'identité et fédération au sein AWS 741
Fédération avec IAM Identiy Center 743
Fédération avec IAM 743
Fédération avec les réserves d'identités Amazon Cognito 744
Ressources supplémentaires 745
Scénarios courants 745

Fédération OIDC 751
Fédération SAML 2.0 778
Fédérer AWS Identités vis-à-vis des services externes 822
Cas d’utilisation courants 823
Comment ça marche 824
Commencer à utiliser la fédération des identités sortantes 824
Comprendre les revendications relatives aux jetons 831
Contrôler l'accès à l'aide de politiques IAM 837
informations d’identification de sécurité temporaires 839
AWS STS and AWS régions 840
Scénarios courants d'informations d'identification temporaires 840
Exemples d'application qui utilisent des informations d'identification temporaires 842
Ressources supplémentaires pour les informations d'identification temporaires 843
Comparez les AWS STS informations d'identification 844
Jeton porteur de service 848
Demande d’identifiants de sécurité temporaires 850
Utilisez des informations d'identification temporaires avec AWS resources 862
Autorisations affectées aux informations d’identification de sécurité temporaires 867
Gérer AWS STS dans un Région AWS 908
AWS STS Régions et points de terminaison 913
Activation de l’accès du fournisseur d’identité personnalisé à la console 919
Balises pour les ressources IAM 935
Choisissez un AWS convention de dénomination des balises 936
Règles de balisage dans IAM et AWS STS 938
Balisage des utilisateurs IAM 940
Blisage de rôles IAM 944
Balisage des politiques gérées par le client 947
Balisage de fournisseurs d’identité OIDC 950
Balisage de fournisseurs d’identité SAML IAM 953
Balisage de profils d'instance 956
Balisage des certificats de serveur 959
Balisage des dispositifs MFA virtuels 961
Marquer les instances du gestionnaire d'accès aux comptes 964
Transmission des balises de session 967
Gestion des accès 984
Ressources de gestion des accès 986

Politiques et autorisations 986
Types de politique 987
Politiques et utilisateur racine 995
Présentation des politiques JSON 995
Accorder le moindre privilège 1000
Politiques gérées et politiques en ligne 1002
Périmètres de données 1014
Limites d'autorisations 1022
Identité et ressource 1037
Contrôle d’accès à l’aide des politiques 1040
Contrôlez l'accès aux utilisateurs et rôles IAM à l'aide de balises 1054
Contrôlez l'accès aux AWS ressources à l'aide de balises 1057
Accès intercompte aux ressources 1062
Transmission des sessions d'accès 1069
Exemples de politiques 1072
Gestion des politiques IAM 1159
Ressources supplémentaires 1160
Création de politiques IAM 1160
Validation de politique 1171
Test de politique 1172
Ajouter ou supprimer des autorisations d'identité 1187
Gestion des versions des politiques IAM 1200
Modification de politiques IAM 1205
Suppression de politiques IAM 1216
Ajustement des autorisations à l’aide des informations consultées 1223
Récapitulatifs de la politique 1816
Récapitulatif de la politique (liste des services) 1817
Niveaux d'accès dans les récapitulatifs de politique 1828
Récapitulatif du service (liste des actions) 1832
Récapitulatif de l'action (liste des ressources) 1838
Exemples de récapitulatifs de la politique 1842
Autorisations requises pour accéder aux autres ressources IAM 1852
Autorisations pour la gestion des identités IAM 1852
Autorisations pour travailler dans Console de gestion AWS 1854
Accordez des autorisations à AWS comptes 1855
Autorisations pour qu'un service accède à un autre service 1856

Actions requises 1856
Exemples de politiques pour IAM 1857
Délégation temporaire 1861
Présentation de 1861
Comment fonctionne la délégation temporaire 1862
Lancer une demande de délégation temporaire 1865
Examiner les demandes de délégation temporaire 1868
Révoquer l'accès temporaire à une délégation 1870
Gestion des autorisations pour les demandes de délégation 1871
Notifications 1872
CloudTrail 1873
Délégation temporaire pour les partenaires 1875
Présentation de 1875
Comment ça marche 1862
Qualification des partenaires 1876
Processus d'intégration 1876
Questionnaire pour partenaires 1877
Comprendre votre intégration 1879
Comprendre les autorisations 1883
Directives d'évaluation des politiques 1885
Modèles de politique 1888
Construisez votre intégration 1899
Gestionnaire d'accès au compte 1908
Qu'est-ce que le gestionnaire d'accès au compte ? 1908
Pourquoi utiliser le gestionnaire d'accès au compte ? 1908
Principales fonctionnalités 1911
Prise en main 1911
Prérequis et considérations 1912
Activer le gestionnaire d'accès au compte 1913
Mettre à jour les pare-feux et les passerelles 1914
Supprimer le gestionnaire d'accès au compte 1916
Gérez l'accès à Comptes AWS 1917
Administration déléguée 1917
Préparez vos rôles IAM 1921
Attribuer et supprimer l'accès 1923
Afficher ou rechercher des devoirs 1927

Sessions et contrôle d'accès 1929
Accès à la main-d'œuvre 1930
Portails utilisateurs 1930
Commencer à utiliser le portail d'accès au compte 1931
Accès utilisateur à Comptes AWS 1932
Sécurité 1938
Gestion des identités et des accès pour le gestionnaire d'accès aux comptes 1938
Protection des données dans le gestionnaire d'accès au compte 1946
Enregistrement et surveillance dans le gestionnaire d'accès aux comptes 1947
Quotas 1947
Quota d'assignation de groupe 1947
Limites d'accélérateur 1947
Exemples de code 1949
IAM 1956
Principes de base 1962
Scénarios 2770
AWS STS 3263
Principes de base 3264
Scénarios 3297
Sécurité 3551
AWS informations d'identification de sécurité 3552
Considérations sur la sécurité 3554
Accès par programmation 3555
AWS directives relatives aux audits de sécurité 3560
Quand effectuer un audit de sécurité ? 3561
Consignes pour l'audit 3561
Vérifiez les informations d'identification AWS de votre compte 3562
Examen de vos utilisateurs IAM 3562
Examen de vos groupes IAM 3563
Examen de vos rôles IAM 3563
Examen de vos fournisseurs IAM pour SAML et OpenID Connect (OIDC) 3563
Examen de vos applications mobiles 3564
Conseils pour l'examen des politiques IAM 3564
Protection des données 3566
Chiffrement des données dans IAM et AWS STS 3567
Gestion des clés dans IAM et AWS STS 3568

Confidentialité du trafic interréseau dans IAM et AWS STS ... 3568
Journalisation et surveillance ... 3568
Enregistrez les événements avec CloudTrail ... 3569
Suivi des tâches privilégiées dans CloudTrail ... 3598
Validation de conformité ... 3602
Résilience ... 3602
Bonnes pratiques pour la résilience IAM ... 3605
Sécurité de l'infrastructure ... 3605
Analyse de la configuration et des vulnérabilités ... 3606
AWS politiques gérées ... 3606
IAMReadOnlyAccess ... 3607
IAMUserChangePassword ... 3607
IAMAccessAnalyzerFullAccess ... 3608
IAMAccessAnalyzerReadOnlyAccess ... 3609
AccessAnalyzerServiceRolePolicy ... 3610
IAMAuditRootUserCredentials ... 3611
IAMCreateRootUserPassword ... 3613
IAMDeleteRootUserCredentials ... 3614
S3UnlockBucketPolicy ... 3616
SQSUnlockQueuePolicy ... 3617
AccountAccessManagerServiceRolePolicy ... 3618
Mises à jour des politiques ... 3618
Fonctions de sécurité en dehors d'IAM ... 3625
Analyseur d'accès IAM ... 3627
Identification des ressources partagées avec une entité externe ... 3627
Identification des accès internes aux ressources critiques pour l'entreprise ... 3630
Identification des accès non utilisés accordés aux utilisateurs et aux rôles IAM ... 3631
Validation des politiques par rapport à AWS bonnes pratiques ... 3631
Validation des politiques par rapport aux normes de sécurité que vous avez spécifiées ... 3632
Générer des politiques ... 3632
Tarification pour l'analyseur d'accès IAM ... 3632
Résultats ... 3633
Fonctionnement des résultats ... 3635
Mise en route avec IAM Access Analyzer ... 3640
Tableau de bord de résultats ... 3659
Examen des résultats ... 3664

Filtrage des résultats 3667
Archiver les résultats 3674
Résolution des problèmes liés aux résultats 3675
Résultats d’erreur 3679
Types de ressources pris en charge 3683
Administrateur délégué 3693
Règles d'archivage 3695
Surveillance avec EventBridge 3698
Intégration à Security Hub CSPM 3710
Se connecter avec CloudTrail 3718
Clés de filtre 3721
Utilisation des rôles liés à un service 3739
Prévisualiser l'accès 3742
Prévisualiser l'accès dans la console Amazon S3 3743
Prévisualisation de l'accès avec IAM Access Analyzer APIs 3744
Vérifications de validation des politiques 3748
Comment fonctionnent les vérifications de politique personnalisées 3749
Exemples de politiques de référence de vérification des nouveaux accès 3750
Inspection des vérifications de politiques personnalisées en échec 3751
Validation à l’aide de vérifications de politique de base 3751
Référence de vérification de politique 3754
Validation à l’aide des vérifications de politiques personnalisées 3901
Génération de la politique de l’analyseur d’accès IAM 3902
Processus de génération de politique 3903
Informations de niveau service et action 3904
À savoir 3904
Autorisations nécessaires 3906
Générer une politique en fonction de CloudTrail l'activité (console) 3909
Générez une politique à l'aide de AWS CloudTrail données d'un autre compte 3913
Générez une politique basée sur CloudTrail l'activité (AWS CLI) 3917
Générez une politique basée sur CloudTrail l'activité (AWS API) 3917
Services de génération d'une politique d'Analyseur d'accès IAM 3918
Quotas de l'IAM Access Analyzer 4477
Résolutions des problèmes IAM 4480
Je n'arrive pas à me connecter à mon AWS compte 4480
J'ai perdu mes clés d'accès 4480

Les variables de la politique ne fonctionnent pas 4481
Les modifications que j'apporte ne sont pas toujours visibles immédiatement 4482
Je ne suis pas autorisé à effectuer : iam : DeleteVirtualMFADevice 4483
Comment créer des utilisateurs IAM en toute sécurité ? 4483
Ressources supplémentaires 4484
Messages d'erreur d'accès refusé 4485
Je reçois un « accès refusé » lorsque je fais une demande à un AWS web 4486
Je reçois un message d'accès refusé lorsque j'effectue une demande avec des informations d'identification de sécurité temporaires 4487
Exemples d'accès refusé 4489
Accès aux messages d'erreur refusés avec ID d'autorisation (aperçu) 4497
Vue d'ensemble 4497
Concepts et définitions 4498
Résoudre les problèmes à l'aide du AWS Console de gestion 4499
Résoudre les problèmes en utilisant AWS API ou AWS CLI 4500
Lorsque vous ne recevez pas d'identifiant d'autorisation 4500
Considérations supplémentaires 4501
Services pris en charge 4501
Problèmes relatifs à l'utilisateur racine 4501
Politiques IAM 4503
Résolution des problèmes à l'aide de l'éditeur visuel 4504
Résolution des problèmes avec les récapitulatifs de politique 4510
Résolution des problèmes de gestion des politiques 4520
Résolution des problèmes de documents de politique JSON 4521
Clés d'accès et clés de sécurité FIDO 4529
Je ne peux pas activer ma clé de sécurité FIDO 4529
Je ne peux pas me connecter à l'aide de ma clé de sécurité FIDO 4530
J'ai perdu ou cassé ma clé de sécurité FIDO 4530
Autres problèmes 4531
Rôles IAM 4531
Je ne parviens pas à endosser un rôle 4531
Un nouveau rôle est apparu dans mon AWS compte 4534
Je ne parviens pas à modifier ou à supprimer un rôle dans mon Compte AWS 4534
Je ne suis pas autorisé à jouer : iam : PassRole 4535
Pourquoi ne puis-je pas assumer un rôle dans le cadre d'une session de 12 heures ? (AWS CLI, AWS API) 4535

Je reçois une erreur lorsque j'essaie de changer de rôle dans la console IAM 4536
Mon rôle dispose d'une politique qui me permet d'effectuer une action, mais j'obtiens « Accès refusé » 4536
Le service n'a pas créé la version de politique par défaut du rôle 4537
Il n'existe aucun cas d'utilisation pour un rôle de service dans la console 4538
IAM et Amazon EC2 4540
Lorsque j'essaie de lancer une instance, je ne vois pas le rôle dans la liste de Rôles IAM de la console Amazon EC2. 4540
Les informations d'identification sur mon instance concernent le mauvais rôle 4541
Quand je tente d'appeler `AddRoleToInstanceProfile`, je reçois un message d'erreur `AccessDenied`. 4541
Amazon EC2 : quand je tente de lancer l'instance avec un rôle, je reçois un message d'erreur `AccessDenied` 4542
Je ne parviens pas à accéder aux informations d'identification de sécurité temporaires sur mon instance EC2 4542
Que signifient les erreurs dans le document `info` de la sous-arborescence IAM ? 4543
IAM et Amazon S3 4545
Comment accorder un accès anonyme à un compartiment Amazon S3 ? 4545
Je suis connecté en tant que Compte AWS utilisateur root. Pourquoi est-ce que je ne parviens pas à accéder à un compartiment Amazon S3 dans mon compte ? 4545
Fédération SAML 2.0 4545
Réponse SAML non valide 4547
RoleSessionName est obligatoire 4547
Non autorisé pour AssumeRoleWithSAML 4548
RoleSessionName Caractères non valides 4549
Caractères d'identité source non valides 4549
Signature de réponse non valide 4549
Clé privée non valide 4550
Échec de la suppression de la clé privée 4550
Impossible de supprimer la clé privée, car l'ID de la clé ne correspond pas à une clé privée 4550
Impossibilité d'endosser un rôle 4550
Impossible d'analyser les métadonnées 4551
Impossible de mettre à jour le fournisseur d'identité 4551
impossible de définir le mode de chiffrement des assertions sur Obligatoire, car aucune clé privée n'est fournie 4551

Impossible d’ajouter et de supprimer des clés privées dans la même requête 4552
Le fournisseur spécifié n'existe pas 4552
DurationSeconds dépasse MaxSessionDuration 4552
La limite de deux clés privées est atteinte 4552
La réponse ne contient pas l'audience requise 4553
Comment IAM travaille avec les autres services AWS 4554
Créez des ressources IAM avec CloudFormation 4554
IAM et modèles CloudFormation 4555
En savoir plus sur CloudFormation 4555
Utilisation AWS CloudShell avec IAM 4556
Obtenir les autorisations IAM pour AWS CloudShell 4556
Interaction avec IAM 4557
Utilisation des AWS SDK 4559
Référence 4561
Identifier AWS ressources avec Amazon Resource Names (ARN) 4561
Format ARN 4561
Rechercher le format d'ARN pour une ressource 4563
Chemins d'accès dans les ARN 4563
Identifiants IAM 4565
Noms conviviaux et chemins 4566
ARN IAM 4567
Identifiants uniques 4574
Présentation des préfixes d'ID uniques 4576
Obtention de l'identifiant unique 4577
IAM et quotas AWS STS 4578
Exigences relatives aux noms IAM 4578
Quotas d'objet IAM 4579
Quotas de l'IAM Access Analyzer 4581
Quotas des rôles Anywhere IAM 4581
Quotas de requête STS 4581
Limites des caractères d'IAM et de STS 4582
Prise en charge des points de terminaison double pile 4587
Prise en charge des points de terminaison double pile IAM 4588
AWS STS prise en charge des terminaux à double pile 4588
Points de terminaison de VPC d’Interface 4589
Disponibilité des points de terminaison d’un VPC 4590

Création d’un point de terminaison de VPC pour IAM 4592
Créez un point de terminaison VPC pour AWS STS 4593
Créez un point de terminaison VPC pour AWS STS Découverte de l'OIDC 4594
Contrôlez l'accès à AWS STS avec des politiques relatives aux terminaux VPC 4597
Utiliser les clés de condition VPC pour contrôler l'accès fédéré 4605
Services qui fonctionnent avec IAM 4612
Services qui fonctionnent avec IAM 4613
En savoir plus 4647
AWS Signature, version 4 4652
Comment ? AWS SiGv4 fonctionne 4653
Comment ? AWS SigV4a fonctionne 4653
Quand signer des demandes ? 4654
Pourquoi les demandes sont-elles signées ? 4655
Ressources supplémentaires 4655
Éléments d’une requête SigV4 4656
Méthodes d’authentification 4659
Création d’une requête signée 4666
Demander des exemples de signature 4680
Résolution des problèmes SigV4 4682
Référence de politique 4687
Références des éléments JSON 4688
Logique d'évaluation de politiques 4804
Syntaxe de politique 4832
AWS politiques gérées pour les fonctions professionnelles 4841
Clés de condition globale 4860
Clés de condition IAM 4948
Actions, ressources et clés de condition 5004
Ressources 5005
Identités 5005
Informations d'identification (mots de passe, clés d'accès et dispositifs MFA) 5005
Autorisations et politiques 5006
Fédération et délégation 5006
IAM et autres AWS produits 5007
Utilisation d'IAM avec Amazon EC2 5007
Utilisation d'IAM avec Amazon S3 5007
Utilisation d'IAM avec Amazon RDS 5008

Utilisation d'IAM avec Amazon DynamoDB 5008
Pratiques de sécurité générales 5008
Ressources générales 5009
Envoi de demandes de requête HTTP 5010
Points de terminaison 5011
HTTPS requis 5011
Signature des demandes d'API IAM 5011
Historique de la documentation 5013
..... 5047

En quoi consiste IAM ?

Follow us on Twitter

Gestion des identités et des accès AWS (IAM) est un service Web qui vous permet de contrôler en toute sécurité l'accès aux AWS ressources. Avec IAM, vous pouvez gérer les autorisations qui contrôlent les AWS ressources auxquelles les utilisateurs peuvent accéder. Vous pouvez utiliser IAM pour contrôler les personnes qui s'authentifient (sont connectées) et sont autorisées (disposent d'autorisations) à utiliser des ressources. IAM fournit l'infrastructure nécessaire au contrôle de l'authentification et des autorisations de votre compte Comptes AWS.

Pour obtenir des instructions sur la configuration Compte AWS et la création de vos identités, consultezMise en route avec IAM.

Identités

Lorsque vous créez un Compte AWS, vous commencez avec une seule identité de connexion appelée utilisateur Compte AWS root qui dispose d'un accès complet à toutes Services AWS les ressources. Il est vivement déconseillé d'utiliser l'utilisateur racine pour vos tâches quotidiennes. Pour les tâches qui requièrent des informations d'identification de l'utilisateur racine, consultez Tâches qui requièrent les informations d'identification de l'utilisateur racine dans le Guide de l'utilisateur IAM.

Utilisez IAM pour configurer d'autres identités en plus de votre utilisateur racine, telles que les administrateurs, les analystes et les développeurs, et leur accorder l'accès aux ressources dont ils ont besoin pour mener à bien leurs tâches.

Gestion des accès

Une fois qu'un utilisateur est configuré dans IAM, il utilise ses informations d'identification de connexion pour s'authentifier auprès d' AWS. L'authentification est assurée en faisant correspondre les informations de connexion à un principal (un utilisateur IAM, un principal AWS STS fédéré, un rôle IAM ou une application) approuvé par le. Compte AWS Ensuite, une demande est effectuée pour accorder au principal l'accès aux ressources. L'accès est accordé en réponse à une demande d'autorisation si l'utilisateur a obtenu une autorisation pour accéder à la ressource. Par exemple, lorsque vous vous connectez à la console pour la première fois et que vous vous trouvez sur sa page d'accueil, vous n'accédez à aucun service spécifique. Lorsque vous sélectionnez un service, la demande d'autorisation est envoyée à ce service et celui-ci vérifie si votre identité figure sur la liste des utilisateurs autorisés, les stratégies appliquées pour contrôler le niveau d'accès accordé et toutes

les autres stratégies susceptibles d'être en vigueur. Les demandes d'autorisation peuvent être faites par des responsables au sein de vous Compte AWS ou par une autre personne en Compte AWS qui vous avez confiance.

Une fois autorisé, le principal peut prendre des mesures ou effectuer des opérations sur les ressources de votre Compte AWS. Par exemple, le principal peut lancer une nouvelle Amazon Elastic Compute Cloud instance, modifier l'appartenance à un groupe IAM ou supprimer des Amazon Simple Storage Service compartiments.

> **Tip**
>
> AWS Skill Builder propose une vidéo de 10 minutes de présentation de l'IAM : Présentation Gestion des identités et des accès AWS sur le site Web de AWS Skill Builder.

Disponibilité du service

L'IAM, comme de nombreux autres AWS services, finit par être cohérent. IAM garantit une haute disponibilité en répliquant les données sur plusieurs serveurs dans les centres de données d'Amazon du monde entier. Si une demande de modification de certaines données aboutit, la modification est validée et stockée en toute sécurité. En revanche, cette modification doit être répliquée dans IAM, ce qui peut prendre un certain temps. Les modifications peuvent être la création ou la mise à jour d'utilisateurs, de groupes, de rôles ou de politiques. Nous vous recommandons de ne pas inclure ce type de modifications IAM dans les chemins de code critique et haute disponibilité de votre application. Au lieu de cela, procédez aux modifications IAM dans une routine d'initialisation ou d'installation distincte que vous exécutez moins souvent. Veillez également à vérifier que les modifications ont été propagées avant que les processus de production en dépendent. Pour de plus amples informations, veuillez consulter Les modifications que j'apporte ne sont pas toujours visibles immédiatement.

Informations sur les coûts des services

Gestion des identités et des accès AWS (IAM) AWS IAM Identity Center et AWS Security Token Service (AWS STS) sont des fonctionnalités de votre AWS compte proposées sans frais supplémentaires. Vous êtes facturé uniquement lorsque vous accédez à d'autres AWS services à l'aide de vos utilisateurs IAM ou de vos identifiants de sécurité AWS STS temporaires.

L'analyseur d'accès IAM propose une analyse des accès externes sans frais supplémentaires. Cependant, des frais vous seront facturés pour les analyses des accès non utilisées et les

vérifications des politiques des clients. Pour une liste complète des frais et des prix pour l'analyseur d'accès IAM, consultez la Tarification de l'analyseur d'accès IAM.

Pour plus d'informations sur la tarification des autres AWS produits, consultez la page de tarification d'Amazon Web Services.

Intégration avec d'autres AWS services

L'IAM est intégré à de nombreux AWS services. Pour obtenir la liste des AWS services qui fonctionnent avec IAM et les fonctionnalités IAM prises en charge par les services, consultez. AWS services qui fonctionnent avec IAM

Pourquoi utiliser IAM ?

Gestion des identités et des accès AWS est un outil puissant pour gérer en toute sécurité l'accès à vos AWS ressources. L'un des principaux avantages de l'utilisation d'IAM est la possibilité d'accorder un accès partagé à votre AWS compte. En outre, IAM vous permet d'attribuer des autorisations granulaires, ce qui vous permet de contrôler exactement les actions que les différents utilisateurs peuvent effectuer sur des ressources spécifiques. Ce niveau de contrôle d'accès est essentiel pour maintenir la sécurité de votre AWS environnement. IAM offre également plusieurs autres fonctionnalités de sécurité. Vous pouvez ajouter l'authentification multifactorielle (MFA) pour un niveau de protection supplémentaire et tirer parti de la fédération d'identité pour intégrer de manière fluide les utilisateurs de votre réseau d'entreprise ou d'autres fournisseurs d'identité. IAM s'intègre également à AWS CloudTrail, fournissant des informations détaillées de journalisation et d'identité pour répondre aux exigences d'audit et de conformité. En tirant parti de ces fonctionnalités, vous pouvez garantir que l'accès à vos AWS ressources critiques est étroitement contrôlé et sécurisé.

Accès partagé à votre Compte AWS

Vous pouvez accorder à d'autres utilisateurs l'autorisation d'administrer et d'utiliser les ressources de votre compte AWS sans avoir à partager votre mot de passe ou clé d'accès.

Autorisations granulaires

Vous pouvez accorder différentes autorisations à différents utilisateurs concernant différentes ressources. Par exemple, vous pouvez autoriser certains utilisateurs à accéder entièrement à Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB, Amazon Redshift et à d'autres services. AWS Pour d'autres utilisateurs, vous pouvez autoriser un accès en lecture seule à certains compartiments Amazon S3 ou l'autorisation

d'administrer certaines instances Amazon EC2 seulement, ou encore un accès à vos informations de facturation uniquement.

Accès sécurisé à AWS ressources pour les applications qui s'exécutent sur Amazon EC2

Vous pouvez utiliser les fonctions IAM pour fournir en toute sécurité des informations d'identification pour les applications s'exécutant sur des instances EC2. Ces informations d'identification permettent à votre application d'accéder à d'autres AWS ressources. Il s'agit notamment des compartiments S3 et des tables DynamoDB.

Multi-factor authentification (MFA)

Vous pouvez ajouter une authentification à deux facteurs à votre compte et aux utilisateurs individuels pour une sécurité renforcée. Avec l'authentification MFA, vos utilisateurs ou vous-même fournissez non seulement un mot de passe ou une clé d'accès pour utiliser votre compte, mais aussi un code généré par un dispositif configuré spécialement. Si vous utilisez déjà une clé de sécurité FIDO avec d'autres services et que sa configuration est AWS prise en charge, vous pouvez l'utiliser WebAuthn pour la sécurité MFA. Pour plus d'informations, consultez Configurations prises en charge pour l'utilisation des clés d'accès et des clés de sécurité

Fédération des identités

Vous pouvez autoriser des utilisateurs disposant déjà de mots de passe ailleurs, par exemple, dans votre réseau d'entreprise ou auprès d'un fournisseur d'identité Internet, d'accéder à votre Compte AWS. Ces utilisateurs reçoivent des informations d'identification temporaires conformes aux recommandations des pratiques exemplaires d'IAM. L'utilisation de la fédération d'identité améliore la sécurité de votre compte AWS .

Informations d'identité par sécurité

Si vous utilisez AWS CloudTrail, vous recevez des enregistrements de journaux incluant des informations sur les utilisateurs effectuant des demandes concernant les ressources de votre compte. Ces informations sont basées sur les identités IAM.

Conformité PCI DSS

IAM prend en charge le traitement, le stockage et la transmission des données de cartes bancaires par un commerçant ou un fournisseur de services et a été validé comme étant conforme à la norme

PCI (Payment Card Industry) DSS (Data Security Standard). Pour plus d'informations sur la norme PCI DSS, notamment sur la procédure à suivre pour demander une copie du package de conformité AWS PCI, consultez la section PCI DSS niveau 1.

Quand utiliser IAM ?

Gestion des identités et des accès AWS est un service d'infrastructure de base qui constitue la base du contrôle d'accès en fonction des identités internes AWS. Vous utilisez IAM chaque fois que vous accédez à votre AWS compte. La façon dont vous utilisez IAM dépend des responsabilités et des fonctions spécifiques au sein de votre organisation. Les utilisateurs des AWS services utilisent IAM pour accéder aux AWS ressources nécessaires à leur travail quotidien, les administrateurs leur accordant les autorisations appropriées. Les administrateurs IAM, quant à eux, sont chargés de gérer les identités IAM et de rédiger des politiques pour contrôler l’accès aux ressources. Quel que soit votre rôle, vous interagissez avec IAM chaque fois que vous authentifiez et autorisez l'accès aux AWS ressources. Cela peut impliquer de se connecter en tant qu’utilisateur IAM, d’assumer un rôle IAM ou de tirer parti de la fédération d’identités pour un accès fluide. Il est essentiel de comprendre les différentes fonctionnalités et cas d'utilisation de l'IAM pour gérer efficacement l'accès sécurisé à votre AWS environnement. Lorsqu’il s’agit de créer des politiques et des autorisations, IAM propose une approche flexible et granulaire. Vous pouvez définir des politiques de confiance pour contrôler quels principaux peuvent endosser un rôle, en plus des politiques basées sur l’identité qui spécifient les actions et les ressources auxquelles un utilisateur ou un rôle peut accéder. En configurant ces politiques IAM, vous pouvez garantir que les utilisateurs et les applications disposent du niveau d’autorisation approprié pour effectuer les tâches requises.

Lorsque vous effectuez différentes activités professionnelles

Gestion des identités et des accès AWS est un service d'infrastructure de base qui constitue la base du contrôle d'accès en fonction des identités internes AWS. Vous utilisez IAM chaque fois que vous accédez à votre compte AWS .

Votre utilisation d'IAM diffère selon le travail que vous effectuez dans AWS.

- Utilisateur du service : si vous utilisez un AWS service pour effectuer votre travail, votre administrateur vous fournit les informations d'identification et les autorisations dont vous avez besoin. Plus vous utilisez de fonctions avancées pour effectuer votre travail, plus vous pouvez avoir besoin d'autorisations supplémentaires. Si vous comprenez bien la gestion des accès, vous saurez demander les autorisations appropriées à votre administrateur.

- Administrateur de services : si vous êtes responsable d'une AWS ressource au sein de votre entreprise, vous disposez probablement d'un accès complet à IAM. Votre responsabilité consiste à déterminer les fonctionnalités IAM ainsi que les ressources auxquelles les utilisateurs de votre service doivent accéder. Vous devez ensuite soumettre les demandes à votre administrateur IAM pour modifier les autorisations des utilisateurs de votre service. Consultez les informations sur cette page pour comprendre les concepts de base d'IAM.
- Administrateur IAM – Si vous êtes administrateur IAM, vous gérez les identités IAM et rédigez des politiques pour gérer l'accès à IAM.

Quand vous êtes autorisé à accéder AWS resources

L'authentification est la façon dont vous vous connectez à AWS l'aide de vos informations d'identité. Vous devez être authentifié en tant qu'utilisateur IAM ou en assumant un rôle IAM. Utilisateur racine d'un compte AWS

Vous pouvez vous connecter en tant qu'identité fédérée à l'aide d'informations d'identification provenant d'une source d'identité telle que AWS IAM Identity Center (IAM Identity Center), de l'authentification unique ou d'informations d'identification. Google/Facebook Pour plus d'informations sur la connexion, consultez Connexion à votre Compte AWS dans le Guide de l'utilisateur Connexion à AWS .

Pour l'accès par programmation, AWS fournit un SDK et une interface de ligne de commande pour signer les demandes de manière cryptographique. Pour plus d'informations, consultez Signature AWS Version 4 pour les demandes d'API dans le Guide de l'utilisateur IAM.

Lorsque vous vous connectez en tant qu'utilisateur IAM

Un utilisateur IAM est une identité qui dispose d'autorisations spécifiques pour une seule personne ou application. Nous vous recommandons d'utiliser ces informations d'identification temporaires au lieu des utilisateurs IAM avec des informations d'identification à long terme. Pour plus d'informations, voir Exiger des utilisateurs humains qu'ils utilisent la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires dans le Guide de l'utilisateur IAM.

Les groupes IAM spécifient une collection d'utilisateurs IAM et permettent de gérer plus facilement les autorisations pour de grands ensembles d'utilisateurs. Pour plus d'informations, consultez Cas d'utilisation pour les utilisateurs IAM dans le Guide de l'utilisateur IAM.

Lorsque vous assumez un rôle IAM

Un rôle IAM est une identité dotée d'autorisations spécifiques qui fournit des informations d'identification temporaires. Vous pouvez assumer un rôle en passant d'un rôle utilisateur à un rôle IAM (console) ou en appelant une opération AWS CLI ou AWS API. Pour plus d'informations, consultez Méthodes pour endosser un rôle dans le Guide de l'utilisateur IAM.

Les rôles IAM sont utiles pour l'accès des utilisateurs fédérés, les autorisations temporaires des utilisateurs IAM, les accès intercompte, les accès entre services et les applications exécutées sur Amazon EC2. Pour plus d'informations, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.

Lorsque vous créez des politiques et des autorisations

Vous accordez des autorisations à un utilisateur en créant une stratégie, qui est un document qui répertorie les actions qu'un utilisateur peut effectuer et les ressources que ces actions peuvent concerner. Les actions ou ressources qui ne sont pas explicitement autorisées sont refusées par défaut. Vous pouvez créer des politiques et les attacher à des principaux (utilisateurs, groupes d'utilisateurs, rôles assumés par des utilisateurs et ressources).

Vous pouvez utiliser ces politiques avec un rôle IAM :

- Politique d'approbation : définit quel principal peut endosser ce rôle et dans quelles conditions. Une politique d'approbation est un type de politique basée sur les ressources pour les rôles IAM. Un rôle ne peut disposer que d'une seule politique d'approbation.
- Identity-based politiques (en ligne et gérées) : ces politiques définissent les autorisations que l'utilisateur du rôle est en mesure d'exécuter (ou dont l'exécution est refusée) et sur quelles ressources.

Utilisez les Exemples de politiques basées sur l'identité IAM pour vous aider à définir des autorisations pour vos identités IAM. Une fois la politique recherchée trouvée, choisissez Afficher la politique pour afficher le JSON de la politique. Vous pouvez utiliser le document de politique JSON sous forme de modèle pour vos propres politiques.

> **Note**
>
> Si vous utilisez IAM Identity Center pour gérer vos utilisateurs, vous attribuez des ensembles d'autorisations dans IAM Identity Center au lieu d'associer une politique d'autorisations à

un principal. Lorsque vous attribuez un ensemble d'autorisations à un groupe ou utilisateur dans le centre d'identité AWS IAM, IAM Identity Center crée les rôles IAM correspondants dans chaque compte et associe les politiques spécifiées dans l'ensemble d'autorisations à ces rôles. IAM Identity Center gère le rôle et permet aux utilisateurs autorisés que vous avez définis d'assumer ce rôle. Si vous modifiez le jeu d'autorisations, IAM Identity Center s'assure que les politiques et les rôles IAM correspondants sont mis à jour en conséquence.
Vous pouvez également utiliser le gestionnaire d'accès aux comptes, une fonctionnalité IAM qui vous permet d'attribuer les rôles IAM existants que vous créez et gérez aux utilisateurs et aux groupes IAM Identity Center, vous donnant ainsi accès à l'ensemble complet des fonctionnalités des rôles IAM. Vous pouvez l'utiliser conjointement avec des ensembles d'autorisations ou seul.
Pour en savoir plus sur IAM Identity Center, consultez Qu’est-ce que IAM Identity Center ? dans le Guide de l’utilisateur AWS IAM Identity Center .

Comment gérer IAM ?

La gestion Gestion des identités et des accès AWS au sein d'un AWS environnement implique l'utilisation d'une variété d'outils et d'interfaces. La méthode la plus courante consiste à utiliser Console de gestion AWS une interface Web qui vous permet d'effectuer un large éventail de tâches administratives IAM, de la création d'utilisateurs et de rôles à la configuration des autorisations.

Pour les utilisateurs plus à l'aise avec les interfaces de ligne de commande, AWS fournit deux ensembles d'outils de ligne de commande : le AWS Command Line Interface et le AWS Tools for Windows PowerShell. Ils vous permettent d’émettre des commandes liées à IAM directement depuis le terminal, souvent de manière plus efficace que de naviguer dans la console. En outre, il vous AWS CloudShell permet d'exécuter des commandes CLI ou SDK directement depuis votre navigateur Web, en utilisant les autorisations associées à votre connexion à la console.

Au-delà de la console et de la ligne de commande, AWS propose des kits de développement logiciel (SDKs) pour différents langages de programmation, vous permettant d'intégrer les fonctionnalités de gestion IAM directement dans vos applications. Vous pouvez également accéder à IAM par programmation à l’aide de l’API Query IAM, qui vous permet d’effectuer des demandes HTTPS directement au service. L’utilisation de ces différentes approches de gestion vous offre la flexibilité nécessaire pour intégrer IAM dans vos flux de travail et processus existants.

Utilisez le Console de gestion AWS

La console AWS de gestion est une application Web qui comprend et fait référence à une vaste collection de consoles de service pour la gestion AWS des ressources. Lors de votre première connexion, vous accédez à la page d’accueil de la console. La page d'accueil permet d'accéder à chaque console de service et propose un emplacement unique pour accéder aux informations nécessaires à l'exécution des tâches AWS connexes. Les services et applications mis à votre disposition une fois connecté à la console dépendent AWS des ressources auxquelles vous êtes autorisé à accéder. Vous pouvez obtenir des autorisations d'accès aux ressources soit en endossant un rôle, soit en étant membre d'un groupe auquel des autorisations ont été accordées, soit en obtenant une autorisation explicite. Pour un compte AWS autonome, l'utilisateur root ou l'administrateur IAM configure l'accès aux ressources. Pour AWS Organizations, le compte de gestion ou l'administrateur délégué configure l'accès aux ressources.

Si vous prévoyez que des personnes utilisent la console de AWS gestion pour gérer les AWS ressources, nous vous recommandons de configurer les utilisateurs avec des informations d'identification temporaires afin de garantir la sécurité. Les utilisateurs IAM qui ont endossé un rôle, les principaux fédérés et les utilisateurs d’IAM Identity Center disposent d’informations d’identification temporaires, tandis que l’utilisateur IAM et l’utilisateur racine disposent d’informations d’identification à long terme. Les informations d'identification de l'utilisateur root fournissent un accès complet à Compte AWS, tandis que les autres utilisateurs disposent d'informations d'identification qui leur permettent d'accéder aux ressources qui leur sont accordées par les politiques IAM.

L'expérience de connexion est différente selon les types d' Console de gestion AWS utilisateurs.

- Les utilisateurs IAM et l'utilisateur root se connectent à partir de l'URL de AWS connexion principale (). https://signin.aws.amazon.com Une fois connectés, ils ont accès aux ressources du compte pour lequel ils ont été autorisés.

  Pour vous connecter en tant qu'utilisateur root, vous devez avoir l'adresse e-mail et le mot de passe de l'utilisateur root.

  Pour vous connecter en tant qu'utilisateur IAM, vous devez disposer du Compte AWS numéro ou de l'alias, du nom d'utilisateur IAM et du mot de passe de l'utilisateur IAM.

  Nous vous recommandons de limiter les utilisateurs IAM de votre compte aux situations spécifiques nécessitant des informations d'identification à long terme, par exemple pour un accès d'urgence, et de n'utiliser l'utilisateur root que pour les tâches nécessitant les informations d'identification de l'utilisateur root.

Pour plus de commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser le nom d'utilisateur IAM et les informations de compte. La prochaine fois que l'utilisateur accède à une page du Console de gestion AWS, la console utilise le cookie pour le rediriger vers la page de connexion au compte.

Déconnectez-vous de la console à la fin de votre session pour empêcher la réutilisation de votre connexion précédente.

- Les utilisateurs d'IAM Identity Center se connectent à l'aide d'un portail d' AWS accès spécifique propre à leur organisation. Une fois connectés, ils peuvent choisir le compte ou l'application auxquels ils veulent accéder. S'ils choisissent d'accéder à un compte, ils choisissent le jeu d'autorisations qu'ils souhaitent utiliser pour la session de gestion.
- Les principaux fédérés OIDC et SAML gérés par un fournisseur d'identité externe lié à un Compte AWS se connectent à l'aide d'un portail d'accès d'entreprise personnalisé. Les ressources AWS disponibles pour les utilisateurs dépendent des politiques sélectionnées par leur organisation.

> **Note**
>
> Pour fournir un niveau de sécurité supplémentaire, l'utilisateur root, les utilisateurs IAM et les utilisateurs d'IAM Identity Center peuvent faire vérifier l'authentification multifactorielle (MFA) AWS avant d'accorder l'accès aux ressources. AWS Lorsque l'authentification MFA est activée, vous devez également avoir accès au périphérique MFA pour vous connecter.

Pour en savoir plus sur la façon dont les différents utilisateurs se connectent à la console de gestion, voir Connexion à la console de AWS gestion dans le Guide de l'utilisateur de AWS connexion.

AWS Outils de ligne de commande

Vous pouvez utiliser les outils de ligne de AWS commande pour émettre des commandes sur la ligne de commande de votre système afin d'exécuter l'IAM et AWS des tâches. L'utilisation de la ligne de commande peut être plus rapide et plus pratique que de la console. Les outils de ligne de commande sont également utiles si vous souhaitez créer des scripts qui exécutent AWS des tâches.

AWS fournit deux ensembles d'outils de ligne de commande : le AWS Command Line Interface(AWS CLI) et le AWS Tools for Windows PowerShell. Pour plus d'informations sur l'installation et l'utilisation du AWS CLI, consultez le guide de AWS Command Line Interface l'utilisateur. Pour plus

d'informations sur l'installation et l'utilisation des outils pour Windows PowerShell, consultez le guide de Outils AWS pour PowerShell l'utilisateur.

Une fois connecté à la console, vous pouvez l'utiliser AWS CloudShell depuis votre navigateur pour exécuter des commandes CLI ou SDK. Les autorisations d'accès aux AWS ressources sont basées sur les informations d'identification que vous avez utilisées pour vous connecter à la console. Selon votre expérience, vous trouverez peut-être que la CLI est une méthode plus efficace pour gérer votre Compte AWS. Pour de plus amples informations, consultez Utilisation AWS CloudShell pour travailler avec AWS Gestion de l’identité et des accès.

AWS Interface de ligne de commande (CLI) et kits de développement logiciel (SDKs)

Les utilisateurs d'IAM Identity Center et d'IAM utilisent différentes méthodes pour authentifier leurs informations d'identification lorsqu'ils s'authentifient via la CLI ou les interfaces d'application (APIs) associées. SDKs

Les informations d'identification et les paramètres de configuration se trouvent à plusieurs endroits, tels que les variables d'environnement système ou utilisateur, les fichiers de AWS configuration locaux, ou sont explicitement déclarés sur la ligne de commande en tant que paramètre. Certains emplacements l’emportent sur d’autres.

IAM Identity Center et IAM fournissent des clés d'accès qui peuvent être utilisées avec la CLI ou le kit SDK. Les clés d'accès IAM Identity Center sont des identifiants temporaires qui peuvent être actualisés automatiquement et leur utilisation est recommandée par rapport aux clés d'accès à long terme associées aux utilisateurs IAM.

Pour gérer votre Compte AWS utilisation de la CLI ou du SDK, vous pouvez utiliser AWS CloudShell votre navigateur. Si vous exécutez CloudShell des commandes CLI ou SDK, vous devez d'abord vous connecter à la console. Les autorisations d'accès aux AWS ressources sont basées sur les informations d'identification que vous avez utilisées pour vous connecter à la console. Selon votre expérience, vous trouverez peut-être que la CLI est une méthode plus efficace pour gérer votre Compte AWS.

Pour le développement d'applications, vous pouvez télécharger la CLI ou le kit SDK sur votre ordinateur et vous connecter à partir de l'invite de commande ou d'une fenêtre Docker. Dans ce scénario, vous configurez les informations d'identification relatives à l'authentification et à l'accès dans le cadre du script CLI ou de l'application SDK. Vous pouvez configurer l'accès programmatique aux ressources de différentes manières, en fonction de l'environnement et de l'accès dont vous disposez.

- Les options recommandées pour authentifier le code local avec le AWS service sont IAM Identity Center et IAM Roles Anywhere.
- Les options recommandées pour authentifier le code exécuté dans un environnement AWS sont d'utiliser des rôles IAM ou des informations d'identification IAM Identity Center.

Lorsque vous vous connectez via le portail AWS d'accès, vous pouvez obtenir des informations d'identification à court terme sur la page d'accueil où vous choisissez votre ensemble d'autorisations. Ces informations d'identification ont une durée définie et ne sont pas actualisées automatiquement. Si vous souhaitez utiliser ces informations d'identification, après vous être connecté au AWS portail, choisissez le, Compte AWS puis choisissez l'ensemble d'autorisations. Sélectionnez Accès par ligne de commande ou par programmation pour afficher les options que vous pouvez utiliser pour accéder aux AWS ressources par programmation ou à partir de la CLI. Pour plus d'informations sur ces méthodes, veuillez consulter la rubrique Getting and refreshing temporary credentials dans le Guide de l'utilisateur IAM Identity Center. Ces informations d'identification sont souvent utilisées lors du développement d'applications pour tester rapidement le code.

Nous vous recommandons d'utiliser les informations d'identification IAM Identity Center qui sont automatiquement actualisées lors de l'automatisation de l'accès à vos AWS ressources. Si vous avez configuré des utilisateurs et des jeux d'autorisations dans IAM Identity Center, utilisez la commande `aws configure sso` pour faire appel à un assistant de ligne de commande qui vous aidera à identifier les informations d'identification disponibles et à les stocker dans un profil. Pour plus d'informations sur la configuration de votre profil, veuillez consulter la rubrique Configure your profile with the `aws configure sso` wizard dans le Guide de l'utilisateur de l'interface de ligne de commande AWS pour la version 2.

> **Note**
>
> De nombreux exemples d'applications utilisent des clés d'accès à long terme associées à des utilisateurs IAM ou à un utilisateur root. N'utilisez les informations d'identification à long terme qu'au sein d'un environnement de test (sandbox), dans le cadre d'un exercice d'apprentissage. Passez en revue les alternatives aux clés d'accès à long terme et planifiez la modification de votre code pour utiliser des informations d'identification alternatives, telles que les informations d'identification IAM Identity Center ou les rôles IAM, dès que possible. Après avoir modifié votre code, supprimez les clés d'accès.

Pour en savoir plus sur la configuration de la CLI, voir Installer ou mettre à jour la dernière version de la AWS CLI dans le Guide de l'utilisateur de l'interface de ligne de AWS commande pour la version 2 et Authentification et informations d'identification d'accès dans le Guide de l'utilisateur de l'interface de ligne de AWS commande

Pour en savoir plus sur la configuration du SDK, consultez les sections Authentication IAM Identity Center dans le guide de référence AWS SDKs and Tools et IAM Roles Anywhere dans le guide de référence AWS SDKs and Tools.

Utilisez le AWS SDKs

AWS fournit SDKs (kits de développement logiciel) composés de bibliothèques et d'exemples de code pour différents langages de programmation et plateformes (Java, Python, Ruby, .NET, iOS, Android, etc.). Ils SDKs fournissent un moyen pratique de créer un accès programmatique à IAM et. AWS Par exemple, ils se SDKs chargent de tâches telles que la signature cryptographique des demandes, la gestion des erreurs et le renouvellement automatique des demandes. Pour plus d'informations AWS SDKs, notamment sur la manière de les télécharger et de les installer, consultez la page Outils pour Amazon Web Services.

Utilisation de l’API Query IAM

Vous pouvez accéder à IAM et par AWS programmation à l'aide de l'API de requête IAM, qui vous permet d'envoyer des requêtes HTTPS directement au service. Lorsque vous utilisez l'API Query , vous devez inclure un code pour signer numériquement les demandes à l'aide de vos informations d'identification. Pour plus d'informations sur les groupes d'utilisateurs, veuillez consulter Appel de l'API IAM à l'aide de requêtes HTTP et la référence API IAM.

Fonctionnement de IAM

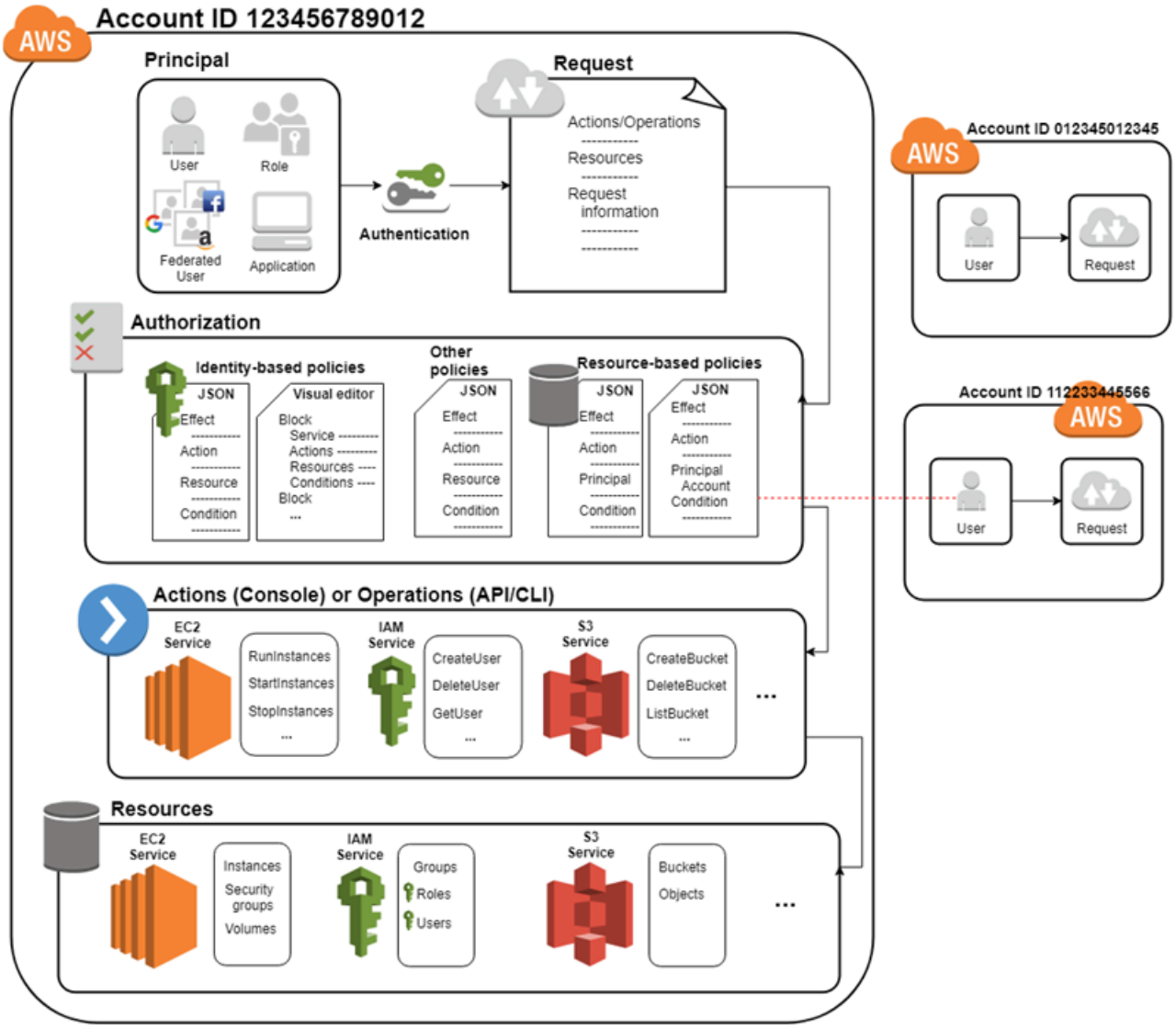
Gestion des identités et des accès AWS fournit l'infrastructure nécessaire pour contrôler l'authentification et l'autorisation de votre Compte AWS.

Tout d'abord, un utilisateur humain ou une application utilise ses informations de connexion pour s'authentifier auprès d' AWS. IAM fait correspondre les informations de connexion à un principal (un utilisateur IAM, un utilisateur principal AWS STS fédéré, un rôle IAM ou une application) approuvé par le principal Compte AWS et authentifie le principal pour y accéder. AWS

Ensuite, IAM effectue une requête pour accorder au principal l’accès aux ressources. IAM accorde ou refuse l’accès en réponse à une requête d’autorisation. Par exemple, lorsque vous vous connectez

à la console pour la première fois et que vous vous trouvez sur sa page d'accueil, vous n'accédez à aucun service spécifique. Lorsque vous sélectionnez un service, vous envoyez une requête d'autorisation à IAM pour ce service. IAM vérifie que votre identité figure sur la liste des utilisateurs autorisés, détermine les politiques qui contrôlent le niveau d'accès accordé et évalue les autres politiques éventuellement en vigueur. Les responsables au sein de vous Compte AWS ou d'un tiers en Compte AWS qui vous avez confiance peuvent faire des demandes d'autorisation.

Une fois autorisé, le principal peut effectuer des actions ou des opérations sur les ressources de votre Compte AWS. Par exemple, le principal peut lancer une nouvelle Amazon Elastic Compute Cloud instance, modifier l'appartenance à un groupe IAM ou supprimer des Amazon Simple Storage Service compartiments. Le diagramme suivant illustre ce processus à travers l'infrastructure IAM :



Composants d’une requête

Lorsqu'un principal essaie d'utiliser l' Console de gestion AWS AWS API, ou le AWS CLI, ce principal envoie une demande à AWS. La demande inclut les informations suivants :

- Actions ou opérations : actions ou opérations que le principal souhaite effectuer, par exemple une action dans l' Console de gestion AWS API ou une opération dans l' AWS API AWS CLI ou.
- Ressources : objet de AWS ressource sur lequel le principal demande d'effectuer une action ou une opération.
- Principal : personne ou application qui utilise une entité (utilisateur ou rôle) pour envoyer la demande. Les informations sur le principal incluent les politiques d’autorisation.
- Données d’environnement : informations sur l’adresse IP, l’agent utilisateur, le statut SSL ou l’horodatage.
- Données sur les ressources : données associées à la ressource demandée, telles que le nom d’une table DynamoDB ou une balise sur une instance Amazon EC2.

AWS rassemble les informations de demande dans un contexte de demande, qu'IAM évalue pour autoriser la demande.

Comment les principaux sont authentifiés

Un principal se connecte à AWS l'aide de ses informations d'identification qu'IAM authentifie pour lui permettre d'envoyer une demande. AWS Certains services, tels qu'Amazon S3 et Amazon AWS STS, autorisent des demandes spécifiques émanant d'utilisateurs anonymes. Cependant, ils sont l’exception à la règle. Chaque type d’utilisateur passe par une authentification.

- Utilisateur root : vos informations de connexion utilisées pour l'authentification sont l'adresse e-mail que vous avez utilisée pour créer Compte AWS et le mot de passe que vous avez spécifié à ce moment-là.
- Principal fédéré : votre fournisseur d'identité vous authentifie et transmet vos informations d'identification à qui AWS vous n'avez pas à vous connecter directement. AWS IAM Identity Center et IAM prennent tous deux en charge la fédération d’identité.
- Utilisateurs inscrits Répertoire AWS IAM Identity Center (non fédérés) : les utilisateurs créés directement dans le répertoire par défaut d'IAM Identity Center se connectent via le portail d' AWS accès et fournissent leur nom d'utilisateur et leur mot de passe.

- Utilisateur IAM : vous vous connectez en fournissant votre ID de compte ou alias, votre nom d'utilisateur et votre mot de passe. Pour authentifier les charges de travail à partir de l'API ou AWS CLI, vous pouvez utiliser des informations d'identification temporaires en assumant un rôle ou vous pouvez utiliser des informations d'identification à long terme en fournissant votre clé d'accès et votre clé secrète.

  Pour en savoir plus sur les entités IAM, consultez Utilisateurs IAM et Rôles IAM.

AWS vous recommande d'utiliser l'authentification multifacteur (MFA) avec tous les utilisateurs afin de renforcer la sécurité de votre compte. Pour en savoir plus sur MFA, consultez AWS Multi-factor authentification dans IAM.

Principes de base des politiques d'autorisation

L'autorisation signifie que le principal dispose des autorisations requises pour traiter sa requête. Lors de l'autorisation, IAM identifie les politiques qui s'appliquent à la requête en utilisant les valeurs du contexte de la requête. Ensuite, il utilise les politiques pour déterminer s'il autorise ou refuse la demande. IAM stocke la plupart des politiques d'autorisation sous forme de documents JSON spécifiant les autorisations pour les entités du principal.

Il existe plusieurs types de politiques pouvant affecter une requête d'autorisation. Pour autoriser vos utilisateurs à accéder aux AWS ressources de votre compte, vous pouvez utiliser des politiques basées sur l'identité. Resource-based les politiques peuvent accorder un accès ??? entre comptes. Pour effectuer une requête dans un autre compte, une politique dans l'autre compte doit vous autoriser à accéder à la ressource et l'entité IAM que vous utilisez pour effectuer la demande doit avoir une politique basée sur une identité qui autorise la requête.

IAM recherche chaque politique qui s'applique au contexte de votre requête. L'évaluation de la politique IAM utilise un refus explicite, ce qui signifie que si une seule politique d'autorisation inclut une action refusée, IAM refuse l'ensemble de la requête et arrête l'évaluation. Les requêtes étant refusées par défaut, les politiques d'autorisation applicables doivent autoriser toutes les parties de votre requête pour qu'IAM l'autorise. La logique d'évaluation pour une requête au sein d'un même compte utilise les règles de base suivantes :

- Par défaut, toutes les demandes sont refusées. (En général, les demandes effectuées à l'aide des informations d'identification du Utilisateur racine d'un compte AWS pour les ressources de ce compte sont toujours autorisées.)

- Une autorisation explicite dans une politique d'autorisations (basée sur l'identité ou les ressources) remplace cette valeur par défaut.
- L'existence d'une politique de contrôle des AWS Organizations services (SCP) ou d'une politique de contrôle des ressources (RCP), d'une limite d'autorisations IAM ou d'une politique de session remplace l'autorisation. S'il existe une ou plusieurs de ces sortes de politiques, elles doivent toutes autoriser la demande. Sinon, elle est refusée implicitement. Pour plus d’informations sur les SCP et les RCP, consultez les Politiques d’autorisation dans AWS Organizations dans le Guide de l’utilisateur AWS Organizations .
- Un refus explicite dans n’importe quelle politique remplace toutes les autorisations dans toutes les politiques.

Pour en savoir plus, veuillez consulter la section Logique d'évaluation de politiques.

Une fois qu’IAM a authentifié et autorisé le principal, IAM approuve les actions ou les opérations figurant dans sa requête en évaluant la politique d’autorisation qui s’applique au principal. Chaque AWS service définit les actions (opérations) qu'il prend en charge et inclut les actions que vous pouvez effectuer sur une ressource, telles que l'affichage, la création, la modification et la suppression de cette ressource. La politique d’autorisation qui s’applique au principal doit inclure les actions nécessaires pour exécuter une opération. Pour en savoir sur l’évaluation des politiques d’autorisation par IAM, consultez the section called “Logique d'évaluation de politiques”.

Le service définit un ensemble d’actions qu’un principal peut effectuer sur chaque ressource. Lorsque vous créez des politiques d’autorisation, veillez à inclure les actions que vous souhaitez que l’utilisateur puisse effectuer. Par exemple, IAM prend en charge plus de 40 actions pour une ressource utilisateur, y compris les actions de base suivantes :

- `CreateUser`
- `DeleteUser`
- `GetUser`
- `UpdateUser`

En outre, vous pouvez spécifier des conditions dans votre politique d’autorisation qui permettent d’accéder aux ressources lorsque la requête remplit les conditions spécifiées. Par exemple, vous pouvez souhaiter qu’une déclaration de politique prenne effet après une date spécifique ou qu’elle contrôle l’accès lorsqu’une valeur spécifique apparaît dans une API. Pour spécifier des conditions, vous utilisez l’élément `Condition` d’une déclaration de politique.

Une fois que l'IAM a approuvé les opérations d'une requête, le principal peut travailler avec les ressources correspondantes de votre compte. Une ressource est une objet existant au sein d'un service. Il peut s'agir, par exemple, d'une instance Amazon EC2, d'un utilisateur IAM et d'un compartiment Amazon S3. Si le principal crée une requête pour effectuer une action sur une ressource qui n'est pas incluse dans la politique d'autorisation, le service refuse la requête. Par exemple, si vous êtes autorisé à supprimer un rôle IAM mais demandez la suppression d'un groupe IAM, la requête échoue si vous n'êtes pas autorisé à supprimer des groupes IAM. Pour en savoir plus sur les actions, les ressources et les clés de condition prises en charge par les différents AWS services, voir Actions, ressources et clés de condition pour les AWS services.

Comparaison des identités et des informations d'identification IAM

Les identités gérées dans Gestion des identités et des accès AWS sont les utilisateurs IAM, les rôles IAM et les groupes IAM. Ces identités s'ajoutent à celles de votre utilisateur root qui AWS a créé avec votre Compte AWS.

Il est vivement recommandé de ne pas utiliser l'utilisateur racine pour vos tâches quotidiennes, y compris pour les tâches administratives. Au lieu de cela, provisionnez des utilisateurs supplémentaires et accordez-leur les autorisations requises pour effectuer les tâches nécessaires. Vous pouvez ajouter des utilisateurs en ajoutant des personnes à votre répertoire IAM Identity Center, en fédérant un fournisseur d'identité externe avec IAM Identity Center ou IAM, ou en créant des utilisateurs IAM à moindre privilège.

Pour plus de sécurité, nous vous recommandons de centraliser l'accès root afin de sécuriser de manière centralisée les informations d'identification de l'utilisateur root de votre utilisation Comptes AWS AWS Organizations gérée. Gestion centralisée de l'accès root pour les comptes membresvous permet de supprimer de manière centralisée et d'empêcher la restauration à long terme des informations d'identification des utilisateurs root, empêchant ainsi tout accès root involontaire à grande échelle. Après avoir activé l'accès racine centralisé, vous pouvez utiliser une session privilégiée pour effectuer des actions sur les comptes membres.

Après avoir configuré vos utilisateurs, vous pouvez donner accès à vos utilisateurs Compte AWS à des personnes spécifiques et leur accorder les autorisations nécessaires pour accéder aux ressources.

À titre de meilleure pratique, il est AWS recommandé de demander aux utilisateurs humains d'assumer un rôle IAM pour accéder AWS afin qu'ils utilisent des informations d'identification temporaires. Si vous gérez des identités dans le répertoire IAM Identity Center ou si vous utilisez la fédération avec un fournisseur d'identité, vous suivez les pratiques exemplaires.

Termes

Ces termes sont couramment utilisés lors de l’utilisation d’identités IAM :

Ressource IAM

Le service IAM stocke ces ressources. Vous pouvez les ajouter, les modifier et les supprimer à partir de la console.

- Utilisateur IAM
- Groupe IAM
- Rôle IAM
- Stratégie d’autorisation
- Identity-provider objet

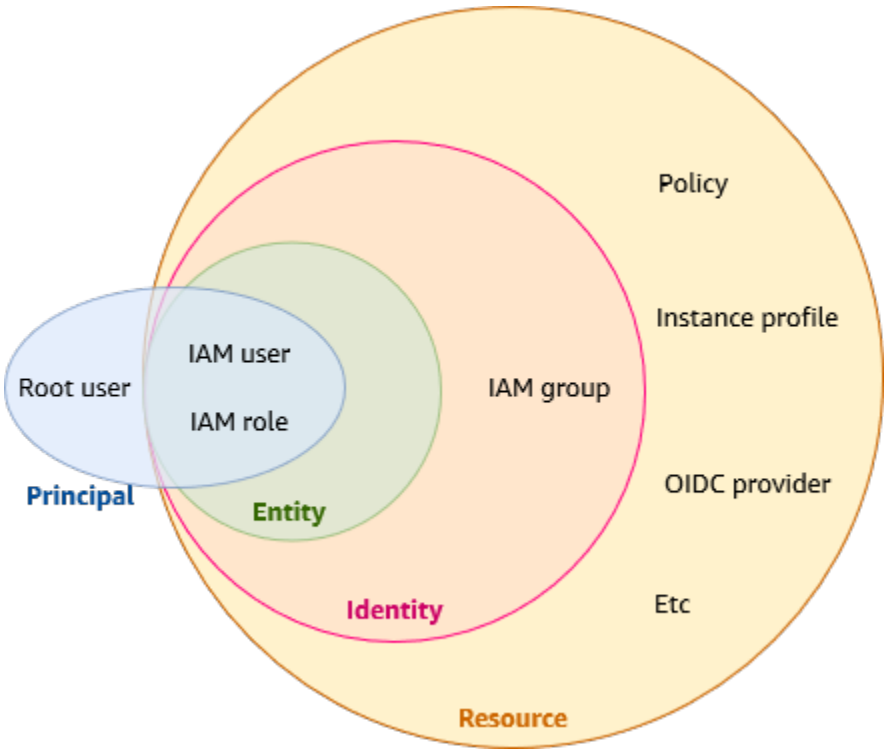
Entité IAM

Ressources IAM AWS utilisées pour l'authentification. Spécifier l’entité en tant que principal dans une politique basée sur les ressources.

- Utilisateur IAM
- Rôle IAM

Identité IAM

La ressource IAM autorisée dans des politiques pour effectuer des actions et accéder aux ressources. Les identités incluent les utilisateurs IAM, les groupes IAM et les rôles IAM.



Principaux

Un Utilisateur racine d'un compte AWS utilisateur IAM ou un rôle IAM qui peut demander une action ou une opération sur une AWS ressource. Les principaux incluent des utilisateurs humains, des charges de travail, des principaux fédérés et des rôles endossés. Après l'authentification, IAM accorde au principal des informations d'identification permanentes ou temporaires auxquelles il peut adresser des demandes AWS, selon le type de principal.

Les utilisateurs humains sont également connus sous le nom d’identités humaines, telles que les personnes, les administrateurs, les développeurs, les opérateurs et les consommateurs de vos applications.

Les charges de travail sont un ensemble de ressources et de codes qui apportent une valeur ajoutée à l’entreprise, tels qu’une application, un processus, des outils opérationnels et d’autres composants.

Les principaux fédérés sont des utilisateurs dont l'identité et les informations d'identification sont gérées par un autre fournisseur d'identité, tel qu'Active Directory, Okta ou Microsoft Entra.

Les rôles IAM sont une identité IAM que vous pouvez créer dans votre compte et qui dispose d’autorisations spécifiques déterminant ce que l’identité peut et ne peut pas faire. En revanche, au lieu d’être associé de manière unique à une personne, un rôle est conçu pour être assumé par tout utilisateur qui en a besoin.

IAM accorde aux utilisateurs IAM et à l'utilisateur root des informations d'identification à long terme et des informations d'identification temporaires aux rôles IAM. utilisateurs dans AWS IAM Identity Center, les principaux fédérés OIDC et SAML assument des rôles IAM lorsqu'ils se connectent AWS, ce qui leur confère des informations d'identification temporaires. Il est recommandé de demander aux utilisateurs humains et aux charges de travail d'accéder aux AWS ressources à l'aide d'informations d'identification temporaires.

Différence entre les utilisateurs IAM et les utilisateurs dans IAM Identity Center

Les utilisateurs IAM ne sont pas des comptes distincts, mais des utilisateurs individuels au sein de votre compte. Chaque utilisateur possède son propre mot de passe pour accéder au Console de gestion AWS. Vous pouvez également créer une clé d'accès individuelle pour chaque utilisateur afin de lui permettre d'effectuer des demandes par programmation en vue d'utiliser des ressources de votre compte.

Les utilisateurs IAM et leurs clés d'accès disposent d'informations d'identification à long terme pour accéder à vos AWS ressources. L'utilisation principale pour les utilisateurs IAM est de donner aux charges de travail qui ne peuvent pas utiliser les rôles IAM la possibilité d'envoyer des requêtes programmatiques aux AWS services à l'aide de l'API ou de l'interface de ligne de commande.

> **Note**
>
> Pour les scénarios dans lesquels vous avez besoin d'utilisateurs IAM disposant d'un accès programmatique et d'informations d'identification à long terme, nous vous recommandons de mettre à jour les clés d'accès. Pour de plus amples informations, veuillez consulter Mise à jour des clés d'accès.

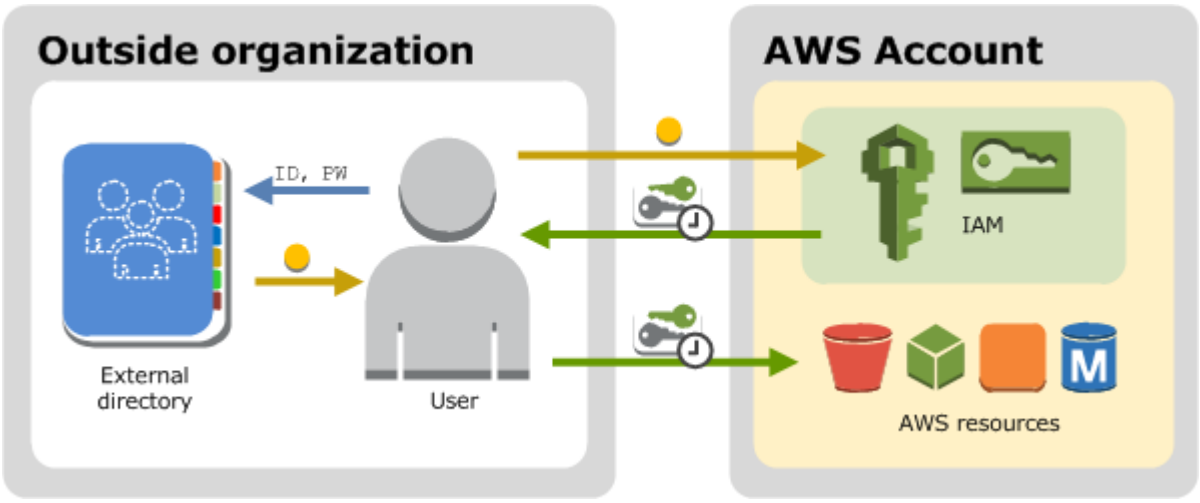
Les identités des employés (personnes) ont utilisateurs dans AWS IAM Identity Center des besoins d'autorisation différents en fonction du rôle qu'ils jouent et peuvent fonctionner dans différents Comptes AWS domaines au sein d'une organisation. Si vous avez des cas d'utilisation qui nécessitent des clés d'accès, vous pouvez les prendre en charge avec utilisateurs dans AWS IAM Identity Center. Les personnes qui se connectent via le portail AWS d'accès peuvent obtenir des clés d'accès avec des informations d'identification à court terme pour accéder à vos AWS ressources. Pour une gestion centralisée des accès, nous vous recommandons d'utiliser AWS IAM Identity Center (IAM Identity Center) pour gérer l'accès à vos comptes et les autorisations au sein de ceux-

ci. IAM Identity Center est automatiquement configuré avec un annuaire Identity Center comme source d'identité par défaut où vous pouvez ajouter des personnes et des groupes et attribuer leur niveau d'accès à vos AWS ressources. Pour plus d'informations, consultez Présentation de AWS IAM Identity Center dans le Guide de l'utilisateur AWS IAM Identity Center .

La principale différence entre ces deux types d'utilisateurs est que les utilisateurs d'IAM Identity Center assument automatiquement un rôle IAM lorsqu'ils se connectent AWS avant d'accéder à la console de gestion ou aux ressources. AWS Les rôles IAM accordent des informations d'identification temporaires à chaque fois que l'utilisateur se connecte. AWS Pour que les utilisateurs IAM puissent se connecter à l'aide d'un rôle IAM, ils doivent être autorisés à assumer et à changer de rôle et ils doivent explicitement choisir de passer au rôle qu'ils souhaitent assumer après avoir accédé au AWS compte.

Fédération des utilisateurs à partir d'une source d'identité existante

Si les utilisateurs de votre organisation sont déjà authentifiés lorsqu'ils se connectent à votre réseau d'entreprise, vous n'avez pas besoin de créer pour eux des utilisateurs IAM distincts ou des utilisateurs dans IAM Identity Center. Au lieu de cela, vous pouvez fédérer ces identités utilisateur en AWS utilisant IAM ou. AWS IAM Identity Center Les principaux fédérés OIDC et SAML assument un rôle IAM qui leur donne l'autorisation d'accéder à des ressources spécifiques. Pour plus d'informations sur les rôles , consultez Termes et concepts relatifs aux rôles.



La fédération est utile dans les cas suivants :

- Vos utilisateurs existent déjà dans un annuaire d'entreprise.

  Si votre annuaire d'entreprise est compatible avec le langage SAML 2.0 (Security Assertion Markup Language 2.0), vous pouvez configurer votre annuaire d'entreprise pour fournir à vos utilisateurs

un accès par authentification unique (SSO). Console de gestion AWS Pour de plus amples informations, veuillez consulter Scénarios courants d'informations d'identification temporaires.

Si votre annuaire d'entreprise n'est pas compatible avec SAML 2.0, vous pouvez créer une application de courtier d'identité pour fournir à vos utilisateurs un accès à connexion unique (SSO). Console de gestion AWS Pour de plus amples informations, veuillez consulter Activez l'accès personnalisé du courtier d'identité au AWS console.

Si votre annuaire d'entreprise est Microsoft Active Directory, vous pouvez l'utiliser AWS IAM Identity Center pour connecter un annuaire autogéré dans Active Directory ou un annuaire AWS Directory Service pour établir la confiance entre votre annuaire d'entreprise et votre Compte AWS.

Si vous utilisez un fournisseur d'identité externe (IdP) tel qu'Okta ou Microsoft Entra pour gérer les utilisateurs, vous pouvez l'utiliser AWS IAM Identity Center pour établir la confiance entre votre IdP et votre. Compte AWS Pour plus d'informations, consultez la section Se connecter à un fournisseur d'identité externe dans le Guide de l'utilisateur AWS IAM Identity Center .

- Vos utilisateurs disposent déjà d'identités Internet.

  Si vous créez une application mobile ou une application basée sur le web permettant aux utilisateurs de s'identifier à l'aide d'un fournisseur d'entité Internet comme Login with Amazon, Facebook, Google ou tout autre fournisseur d'identité compatible avec OpenID Connect (OIDC), l'application peut utiliser la fédération pour accéder à AWS. Pour de plus amples informations, veuillez consulter Fédération OIDC.

  > **Tip**
  >
  > Pour utiliser la fédération d'identité avec des fournisseurs d'identité, nous vous recommandons d'utiliser Amazon Cognito.

Différentes méthodes pour fournir un accès aux utilisateurs

Voici comment vous pouvez donner accès à vos AWS ressources.

Type d'accès utilisateur	Quand est-elle utilisée ?	Comment puis-je obtenir plus d'informations ?
Accès par authentification unique pour les personnes, tels que les utilisateurs de votre personnel, aux ressources AWS à l'aide d'IAM Identity Center	IAM Identity Center constitue un endroit central qui regroupe l'administration des utilisateurs, leur accès aux applications cloud Comptes AWS et leur accès. Vous pouvez configurer un magasin d'identités dans IAM Identity Center ou configurer une fédération avec un fournisseur d'identité (IdP) existant. Les meilleures pratiques de sécurité recommandent d'accorder à vos utilisateurs humains des informations d'identification limitées pour accéder aux AWS ressources. Les personnes bénéficient d'une expérience de connexion simplifiée et vous conservez le contrôle de leur accès aux ressources à partir d'un système unique. IAM Identity Center prend en charge l'authentification multifactorielle (MFA) pour renforcer la sécurité des comptes.	Pour plus d'informations sur la configuration d'IAM Identity Center, consultez Mise en route dans le Guide de l'utilisateur AWS IAM Identity Center . Pour plus d'informations sur l'utilisation de l'authentification multifacteur dans IAM Identity Center, consultez la section Multi-factor Authentification dans le Guide de l'AWS IAM Identity Center utilisateur.
Accès fédéré pour les	Supports IAM IdPs compatibles avec OpenID Connect	Pour plus d'informations sur les fournisseurs d'identité IAM et la fédération, consultez

Type d'accès utilisateur	Quand est-elle utilisée ?	Comment puis-je obtenir plus d'informations ?
utilisateurs humains, tels que les utilisateurs de votre personnel , aux AWS services utilisant des fournisseurs d'identité IAM () IdPs	(OIDC) ou SAML 2.0 (Security Assertion Markup Language 2.0). Après avoir créé un fournisseur d'identité IAM, vous créez un ou plusieurs rôles IAM qui peuvent être attribués de manière dynamique à un principal fédéré.	Fournisseurs d'identité et fédération au sein AWS.
Cross-account accès entre Comptes AWS	Vous souhaitez partager l'accès à certaines AWS ressources avec des utilisateurs d'autres ressources Comptes AWS. Les rôles constituent le principal moyen d'accorder l'accès intercompte. Toutefois , certains services AWS prennent en charge les politiques basées sur les ressources, ce qui vous permet d'attacher une politique directement à une ressource (au lieu d'utiliser un rôle en tant que proxy).	Pour plus d'informations sur les rôles IAM, consultez Rôles IAM. Pour plus d'informations sur les rôles liés à un service, consultez Créer un rôle lié à un service. Pour connaître les services qui prennent en charge l'utilisation de rôles liés à un service, consultez AWS services qui fonctionnent avec IAM. Trouvez les services pour lesquels la case Oui est cochée dans la colonne Service-Linked Rôle. Pour consulter la documentation relative au rôle lié à un service, sélectionnez le lien associé à Oui dans cette colonne.

Type d'accès utilisateur	Quand est-elle utilisée ?	Comment puis-je obtenir plus d'informations ?
Long-term informations d'identification pour les utilisateurs IAM désignés dans votre Compte AWS	Vous pouvez avoir des cas d'utilisation spécifiques nécessitant des informations d'identification à long terme auprès des utilisateurs IAM. AWS Vous pouvez utiliser IAM pour créer ces utilisateurs IAM dans votre Compte AWS ordinateur et utiliser IAM pour gérer leurs autorisations. Voici certaines des fonctionnalités les plus utilisées : • Charges de travail qui ne peuvent pas utiliser des rôles IAM • Third-party AWS clients nécessitant un accès programmatique via des clés d'accès • Service-specific informations d'identification pour AWS CodeCommit Amazon Keyspaces • AWS IAM Identity Center n'est pas disponible pour votre compte et vous n'avez aucun autre fournisseur d'identité En guise de bonne pratique pour les scénarios dans lesquels vous avez besoin	Pour plus d'informations sur la configuration d'un utilisateur IAM, consultez Créez un utilisateur IAM dans votre Compte AWS. Pour plus d'informations sur les clés d'accès des utilisateurs IAM, consultez Gestion des clés d'accès pour les utilisateurs IAM. Pour plus d'informations sur les informations d'identification spécifiques au service AWS CodeCommit ou Amazon Keyspaces, consultez et. Informations d'identification IAM pour CodeCommit : informations d'identification Git, clés SSH et AWS clés d'accès Utilisation d'IAM avec Amazon Keyspaces (pour Apache Cassandra)

Type d'accès utilisateur	Quand est-elle utilisée ?	Comment puis-je obtenir plus d’informations ?
	d'utilisateurs IAM disposant d'un accès programmatique et d'informations d'identification à long terme, nous vous recommandons d'effectuer une mise à jour des clés d'accès. Pour de plus amples informations, veuillez consulter Mise à jour des clés d’accès.	

Prise en charge de l’accès programmatique des utilisateurs

Les utilisateurs ont besoin d'un accès programmatique s'ils souhaitent interagir avec des AWS personnes extérieures au Console de gestion AWS. La manière d'accorder l'accès par programmation dépend du type d'utilisateur qui accède AWS :

- Si vous gérez les identités dans IAM Identity Center, les AWS API nécessitent un profil, ainsi qu'un profil ou une variable d'environnement. AWS Command Line Interface
- Si vous avez des utilisateurs IAM, les AWS API et les clés AWS Command Line Interface d'accès requises. Lorsque cela est possible, créez des informations d’identification temporaires composées d’un ID de clé d’accès, d’une clé d’accès secrète et d’un jeton de sécurité qui indique la date d’expiration des informations d’identification.

Pour accorder aux utilisateurs un accès programmatique, choisissez l’une des options suivantes.

Quel utilisateur a besoin d’un accès programmatique ?	Option	En savoir plus
Identités de la main-d’œuvre (Personnes et utilisateurs gérés dans IAM Identity Center)	Utilisez des informations d'identification à court terme pour signer des demandes programmatiques adressées aux AWS API AWS CLI or	Pour cela AWS CLI, suivez les instructions de la section Obtention des informations d'identification du rôle IAM pour l'accès à l'interface de

Quel utilisateur a besoin d'un accès programmatique ?	Option	En savoir plus
	(directement ou à l'aide des AWS kits de développement logiciel).	ligne de commande du Guide de l'AWS IAM Identity Center utilisateur. Pour les AWS API, suivez les instructions figurant dans les informations d'identification SSO du Guide de référence des AWS kits de développe ment logiciel et des outils.
Utilisateurs IAM	Utilisez des informations d'identification à court terme pour signer des demandes programmatiques adressées aux AWS API AWS CLI or (directement ou à l'aide des AWS kits de développement logiciel).	Suivez les instructions de la section Utilisation d'informa tions d'identification temporair es avec AWS les ressources.
Utilisateurs IAM	Utilisez des informations d'identification à long terme pour signer des demandes programmatiques adressées aux AWS API AWS CLI or (directement ou à l'aide des AWS kits de développement logiciel). (Non recommandé)	Suivez les instructions de la section Gestion des clés d'accès pour les utilisateurs IAM.

Quel utilisateur a besoin d'un accès programmatique ?	Option	En savoir plus
Principaux fédérés	Utilisez une opération d'API AWS STS pour créer une nouvelle session avec des informations d'identification de sécurité temporaires qui incluent une paire de clés d'accès et un jeton de session.	Pour obtenir des explications sur les opérations d'API, consultez the section called "Demande d'identifiants de sécurité temporaires"

Comment les autorisations et les politiques assurent la gestion des accès

La partie de gestion des accès de Gestion des identités et des accès AWS (IAM) vous aide à définir ce qu'une entité principale peut faire dans un compte. Une entité de principal est une personne ou une application qui est authentifiée à l'aide d'une entité IAM (utilisateur ou rôle IAM). La gestion des accès est souvent appelée autorisation. Vous gérez l'accès en AWS créant des politiques et en les associant à des identités ou à des ressources IAM (utilisateurs IAM, groupes IAM ou rôles IAM). AWS Une politique est un objet AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit leurs autorisations. AWS évalue ces politiques lorsqu'un principal utilise une entité IAM (utilisateur IAM ou rôle IAM) pour faire une demande. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée. La plupart des politiques sont stockées AWS sous forme de documents JSON. Pour plus d'informations sur les types de politiques et les utilisations, veuillez consulter Politiques et autorisations dans Gestion des identités et des accès AWS.

Politiques et comptes

Si vous gérez un seul compte dans AWS, vous définissez les autorisations au sein de ce compte à l'aide de politiques. Si vous gérez des autorisations sur plusieurs comptes, la gestion des autorisations pour vos utilisateurs IAM est plus compliquée. Vous pouvez utiliser des rôles IAM, des politiques basées sur les ressources ou des listes de contrôle d'accès (ACLs) pour les autorisations entre comptes. Toutefois, si vous possédez plusieurs comptes, nous vous recommandons plutôt d'utiliser le AWS Organizations service pour vous aider à gérer ces autorisations. Pour plus d'informations, voir Qu'est-ce que c'est AWS Organizations ? dans le guide de AWS Organizations l'utilisateur.

Politiques et utilisateurs

Les utilisateurs IAM sont des identités dans Compte AWS. Lorsque vous créez un utilisateur IAM, il ne peut accéder à rien dans votre compte tant que vous ne lui en donnez pas l'autorisation. Vous accordez des autorisations à un utilisateur IAM en créant une politique basée sur l'identité, qui est attachée à l'utilisateur IAM ou à un groupe IAM auquel l'utilisateur IAM appartient. L'exemple suivant illustre une politique JSON qui autorise l'utilisateur IAM à exécuter toutes les actions Amazon DynamoDB (`dynamodb:*`) sur le tableau `Books` dans le compte `123456789012` de la région `us-east-2`.

JSON

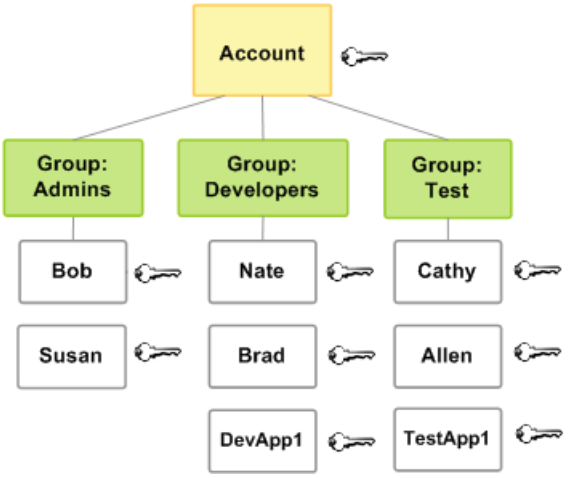
```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
    "Resource": "arn:aws:dynamodb:us-east-2:123456789012:table/Books"
  }
}
```

Une fois cette politique attachée à votre utilisateur IAM, ce dernier est autorisé à exécuter toutes les actions du tableau `Books` de votre instance DynamoDB. La plupart des utilisateurs IAM ont plusieurs politiques qui se combinent pour représenter le total des autorisations qui leur ont été accordées.

Les actions ou ressources qui ne sont pas autorisées explicitement par une politique sont refusées par défaut. Par exemple, si la politique précédente est la seule politique attachée à un utilisateur, alors cet utilisateur peut effectuer des actions DynamoDB sur la table `Books`, mais ne peut pas effectuer d'actions sur d'autres tables. De même, l'utilisateur n'est pas autorisé à effectuer des actions dans Amazon EC2, Amazon S3 ou dans tout autre AWS service, car les autorisations d'utilisation de ces services ne sont pas incluses dans la politique.

Politiques et groupes IAM

Vous pouvez organiser les utilisateurs IAM en groupes IAM et attacher une politique à un groupe IAM. Dans ce cas, les utilisateurs IAM disposent toujours de leurs propres informations d'identification, mais tous les utilisateurs IAM d'un groupe IAM détiennent les autorisations attachées à ce groupe IAM. Utilisez des groupes IAM pour faciliter la gestion des autorisations.



Plusieurs politiques accordant différentes autorisations peuvent être attachées à des utilisateurs IAM ou des groupes IAM. Dans ce cas, la combinaison des politiques détermine les autorisations effectives du principal. Si le principal n'a pas d'`Allow`autorisation explicite pour une action et une ressource, il ne dispose pas de ces autorisations.

sessions d'utilisateur fédéré et rôle

Les principaux fédérés n'ont pas d'identité permanente comme Compte AWS le font les utilisateurs IAM. Pour attribuer des autorisations à des principaux fédérés, vous pouvez créer une entité appelée rôle et définir des autorisations pour le rôle. Lorsqu'un principal fédéré SAML ou OIDC se connecte AWS, l'utilisateur est associé au rôle et reçoit les autorisations définies dans le rôle. Pour de plus amples informations, veuillez consulter Créer un rôle pour un fournisseur d'identité tiers.

Politiques basées sur les ressources et politiques basées sur l'identité

Les politiques basées sur l'identité sont des politiques d'autorisations que vous pouvez attacher à une identité IAM, tel qu'un utilisateur, un groupe ou un rôle IAM. Les politiques basées sur les ressources sont des politiques d'autorisations que vous attachez à une ressource, telle qu'un compartiment Amazon S3 ou une politique d'approbation de rôle IAM.

Les politiques basées sur l'identité contrôlent les actions que peut effectuer l'identité, sur quelles ressources et dans quelles conditions. Les politiques basées l'identité peuvent être classées comme suit :

- Politiques gérées : politiques autonomes basées sur l'identité que vous pouvez associer à plusieurs utilisateurs, groupes et rôles au sein de votre. Compte AWS Vous pouvez utiliser deux types de politiques gérées :

- AWS politiques gérées : politiques gérées créées et gérées par AWS. Si vous utilisez des politiques pour la première fois, nous vous recommandons de commencer par utiliser des politiques AWS gérées.
- Politiques gérées par le client : politiques gérées que vous créez et gérez dans votre Compte AWS. Les politiques gérées par le client fournissent un contrôle plus précis de vos politiques que les politiques AWS gérées. Vous pouvez créer, modifier et valider une politique IAM dans l'éditeur visuel ou en créant le document de politique JSON directement. Pour plus d’informations, consultez Définition d’autorisations IAM personnalisées avec des politiques gérées par le client et Modification de politiques IAM.

- Politiques en ligne : politiques que vous créez et gérez, et qui sont intégrées directement à un utilisateur, groupe ou rôle. Dans la plupart des cas, nous vous déconseillons d'utiliser des politiques en ligne.

Les politiques basées sur les ressources contrôlent les actions qu'un principal spécifique peut effectuer sur cette ressource et dans quelles conditions. Les politiques basées sur les ressources sont des politiques en ligne et il n'y a pas de politiques basées sur des ressources gérées. Pour permettre un accès intercompte, vous pouvez spécifier un compte entier ou des entités IAM dans un autre compte en tant que principal dans une politique basée sur les ressources.

Le service IAM prend en charge un type de politique basée sur les ressources, nommé politique d’approbation de rôle, que vous attachez à un rôle IAM. Étant donné qu’un rôle IAM est à la fois une identité et une ressource qui prend en charge les politiques basées sur les ressources, vous devez associer une politique d’approbation et une politique basée sur une identité à un rôle IAM. Les politiques de confiance définissent les entités principales (comptes, utilisateurs, rôles et utilisateurs principaux AWS STS fédérés) qui peuvent assumer le rôle. Pour en savoir plus sur la façon dont les rôles IAM sont différents d'autres politiques basées sur les ressources, veuillez consulter Accès intercompte aux ressources dans IAM.

Pour connaître les services qui prennent en charge les politiques basées sur les ressources, voir AWS services qui fonctionnent avec IAM. Pour en savoir plus sur politiques basées sur les ressources, voir Identity-based politiques et politiques basées sur les ressources.

Définition des autorisations en fonction des attributs avec autorisation ABAC

Attribute-based le contrôle d'accès (ABAC) est une stratégie d'autorisation qui définit les autorisations en fonction des attributs. AWS appelle ces balises d'attributs. Vous pouvez associer des balises aux ressources IAM, notamment aux entités IAM (utilisateurs IAM ou rôles IAM) et aux ressources. AWS Vous pouvez créer une seule politique ABAC ou un petit nombre de politiques pour vos principaux IAM. Vous pouvez concevoir des politiques ABAC qui autorisent les opérations lorsque la balise du principal correspond à la balise de la ressource. Le système d'attributs d'ABAC fournit à la fois un contexte utilisateur élevé et un contrôle d'accès granulaire. ABAC étant basé sur les attributs, il peut effectuer une autorisation dynamique pour les données ou les applications qui accorde ou révoque l'accès en temps réel. ABAC est utile dans les environnements évolutifs et dans les situations où la gestion des politiques d'identité ou de ressources est devenue complexe.

Par exemple, vous pouvez créer trois rôles IAM avec la clé de balise `access-project`. Définissez la valeur de la balise du premier rôle IAM sur `Heart`, celle du deuxième sur `Star`, et celle du troisième sur `Lightning`. Vous pouvez ensuite utiliser une politique unique qui autorise l'accès lorsque le rôle IAM et la AWS ressource ont la valeur `access-project` de balise. Un didacticiel détaillé expliquant comment utiliser l'ABAC dans AWS est disponible à la section Tutoriel IAM : définir les autorisations d'accès AWS ressources basées sur des balises. Pour en savoir plus sur les services qui prennent en charge ABAC, consultez AWS services qui fonctionnent avec IAM.
Ce schéma montre que les balises appliquées à un principal doivent correspondre aux balises appliquées à une ressource pour que l'utilisateur obtienne des autorisations sur la ressource. Les balises doivent être appliquées aux groupes IAM, aux groupes de ressources, aux utilisateurs individuels et aux ressources individuelles.

Comparaison d'ABAC et du modèle RBAC traditionnel

Le modèle d'autorisation traditionnel utilisé dans IAM est le contrôle d'accès basé sur les rôles (RBAC). RBAC définit les autorisations en fonction de la tâche ou du rôle d'une personne, ce qui est différent d'un rôle IAM. IAM inclut des politiques gérées pour des fonctions de tâches, qui alignent les autorisations sur une fonction de tâche dans un modèle RBAC.

Dans IAM, vous implémentez le RBAC en créant différentes politiques pour différentes activités professionnelles. Vous attachez ensuite les politiques à des identités (utilisateurs, groupes ou rôles IAM). Les bonnes pratiques consistent à n'octroyer que les autorisations minimales nécessaires pour la tâche à accomplir. Cela se traduit par un accès avec le moindre privilège. Chaque politique de

fonction de travail répertorie les ressources spécifiques auxquelles les identités assignées à cette politique peuvent accéder. L’inconvénient du modèle RBAC traditionnel est que lorsque vous ou vos utilisateurs ajoutez de nouvelles ressources à votre environnement, vous devez mettre à jour les politiques pour permettre l’accès à ces ressources.

Par exemple, supposons que vous ayez trois projets, nommés `Heart`, `Star` et `Lightning`, sur lesquels travaillent vos employés. Vous créez un rôle IAM pour chaque projet. Vous attachez ensuite des politiques à chaque rôle IAM pour définir les ressources auxquelles toute personne autorisée à endosser le rôle IAM peut accéder. Si un employé change de poste au sein de votre entreprise, vous lui assignez un autre rôle IAM. Vous pouvez attribuer des personnes ou des programmes à plusieurs rôles IAM. Toutefois, le projet `Star` peut nécessiter des ressources supplémentaires, telles qu'un nouveau conteneur Amazon EC2. Dans ce cas, vous devez mettre à jour la politique attachée au rôle IAM `Star` pour spécifier la nouvelle ressource du conteneur. Sinon, les membres du projet `Star` ne seront pas autorisés à accéder au nouveau conteneur.
Ce diagramme illustre le fait que le contrôle d’accès basé sur les rôles exige que chaque identité se voie attribuer une politique spécifique basée sur la fonction de la tâche pour accéder aux différentes ressources.

L'ABAC offre les avantages suivants par rapport au modèle RBAC traditionnel

- Les autorisations de l'ABAC sont évolutives. L'administrateur n'a plus besoin de mettre à jour les politiques existantes pour autoriser l'accès aux nouvelles ressources. Par exemple, supposons que vous ayez conçu votre stratégie ABAC avec la balise `access-project` Un développeur utilise le rôle IAM avec la balise `access-project` = `Heart`. Lorsque des membres du projet `Heart` ont besoin de ressources Amazon EC2 supplémentaires, le développeur peut créer des instances Amazon EC2 avec la balise `Heart` = `access-project`. N'importe quel membre du projet `Heart` peut alors démarrer et arrêter ces instances, car leurs valeurs de balise correspondent.
- L'ABAC exige moins de politiques. Comme vous n'avez pas besoin de créer une politique pour chaque activité professionnelle, vous créez moins de politiques. Leur gestion s'en trouve simplifiée.
- Grâce à l’ABAC, les équipes peuvent réagir de manière dynamique aux changements et à la croissance. Comme les autorisations pour les nouvelles ressources sont automatiquement accordées en fonction des attributs, il n’est pas nécessaire d’attribuer manuellement des politiques aux identités. Par exemple, si votre entreprise prend déjà en charge les projets `Heart` et `Star` avec l'ABAC, il est facile d'ajouter un nouveau projet `Lightning`. Un administrateur IAM crée un rôle IAM avec la balise `access-project` = `Lightning`. Il n'est pas nécessaire de modifier la politique pour prendre en charge un nouveau projet. Toute personne autorisée à endosser le rôle IAM peut créer et afficher des instances balisées avec `access-project` = `Lightning`. Un autre

scénario est celui où un membre de l'équipe passe du projet `Heart` au projet `Lightning`. Pour permettre aux membres de l'équipe d'accéder au projet `Lightning`, l'administrateur IAM leur attribue un rôle IAM différent. Il n'est pas nécessaire de modifier les politiques d'autorisations.

- Les autorisations détaillées sont possibles avec l'ABAC. Lorsque vous créez des politiques, la bonne pratique consiste à accorder un moindre privilège. Avec le modèle RBAC traditionnel, vous écrivez une politique qui autorise l'accès à des ressources spécifiques uniquement. Avec l'ABAC, vous pouvez autoriser des actions sur toutes les ressources si les balises de la ressource et du principal correspondent.
- Utilisez les attributs des employés figurant dans l'annuaire de votre entreprise avec l'ABAC. Vous pouvez configurer votre fournisseur SAML ou OIDC pour qu'il transmette les balises de session à IAM. Lorsque vos employés se fédérent dans AWS, IAM applique leurs attributs au principal qui en résulte. Vous pouvez ensuite utiliser l'ABAC pour autoriser ou refuser des autorisations sur la base de ces attributs.

Pour un didacticiel détaillé expliquant comment utiliser ABAC dans AWS, voirTutoriel IAM : définir les autorisations d'accès AWS ressources basées sur des balises.

Mise en route avec IAM

Gestion des identités et des accès AWS (IAM) vous permet de contrôler en toute sécurité l'accès à Amazon Web Services (AWS) et aux ressources de votre compte. IAM peut également préserver la confidentialité de vos informations d'identification. Vous ne vous inscrivez pas spécifiquement pour utiliser IAM. L'utilisation d'IAM n'induit aucuns frais.

La façon dont vous utilisez IAM dépend de votre fonction professionnelle. Consultez Quand utiliser IAM ? les conseils spécifiques à chaque rôle.

Utilisez IAM pour donner aux identités, telles que les utilisateurs et les rôles, accès aux ressources de votre compte. Par exemple, vous pouvez utiliser IAM avec des utilisateurs existants de votre annuaire d'entreprise que vous gérez en externe AWS ou vous pouvez créer des utilisateurs en AWS utilisant AWS IAM Identity Center. Les identités fédérées endossent des rôles IAM définis pour accéder aux ressources dont elles ont besoin. Pour en savoir plus sur IAM Identity Center, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

> **Note**
>
> L'IAM est intégré à plusieurs AWS produits. Pour connaître la liste des services prenant en charge IAM, veuillez consulter AWS services qui fonctionnent avec IAM.

Pour en savoir plus sur la mise en route AWS, la création d'un AWS Organizations utilisateur administratif et l'utilisation de plusieurs services pour résoudre un problème tel que la création et le lancement de votre premier projet, consultez le Centre de ressources Getting Started.

Pour vous familiariser avec les tâches IAM courantes, consultezDidacticiels IAM. Les tâches incluent la création de rôles pour l'accès entre comptes et l'utilisation de balises pour le contrôle d'accès basé sur les attributs.

Après avoir configuré votre compte, planifiez la manière dont les utilisateurs et les charges de travail y accèdent. Pour de plus amples informations, veuillez consulter Planifiez l'accès à votre AWS compte.

Utilisez ensuite Préparation des autorisations de moindre privilège le principe du moindre privilège.

Configuration de votre Compte AWS

Avant de commencer à travailler avec IAM, assurez-vous d'avoir terminé la configuration initiale de votre AWS environnement.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. Vous pouvez afficher l'activité en cours de votre compte et gérer votre compte à tout moment en accédant à https://aws.amazon.com/ et en cliquant sur Mon compte.

Lorsque vous vous êtes inscrit au service, vous en avez créé un à Compte AWS l'aide d'une adresse e-mail et d'un mot de passe. Il s'agit de vos informations d'identification d'utilisateur AWS root. Il est recommandé de ne pas utiliser vos informations d'identification d'utilisateur root AWS pour accéder aux tâches quotidiennes. N'utilisez vos informations d'identification d'utilisateur racine que pour effectuer des tâches nécessitant des informations d'identification d'utilisateur racine. De même, ne partagez vos informations d'identification avec personne. Ajoutez plutôt des personnes à votre annuaire et donnez-leur accès à votre Compte AWS.

Pour sécuriser votre Utilisateur racine d'un compte AWS

1. Connectez-vous en Console de gestion AWS tant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l'aide pour vous connecter en utilisant l'utilisateur racine, consultez Connexion en tant qu'utilisateur racine dans le Guide de l'utilisateur Connexion à AWS .
2. Activez l'authentification multifactorielle (MFA) pour votre utilisateur racine.

   Pour obtenir des instructions, consultez la section Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le Guide de l'utilisateur IAM.

Octroi d'accès à la console de facturation

Les utilisateurs et les rôles IAM dans un Compte AWS ne peuvent pas accéder à la console de facturation et gestion des coûts. Cela est vrai même s'ils disposent de politiques IAM qui accordent l'accès à certaines fonctions de facturation. Pour accorder l'accès, l'utilisateur racine de l'Compte AWS doit d'abord activer l'accès IAM.

> **Note**
>
> En tant que pratiques exemplaires en matière de sécurité, nous vous recommandons de fournir l'accès à vos ressources par le biais d'une fédération d'identité avec AWS IAM Identity Center. Lorsque vous activez IAM Identity Center conjointement avec AWS Organizations, la console de gestion de la facturation et des coûts est activée par défaut avec une facturation consolidée pour tous les membres Comptes AWS de votre organisation. Pour plus d'informations sur la facturation consolidée, consultez Consolidation de la facturation pour AWS Organizations dans le Guide de l'utilisateur de la Facturation et gestion des coûts.

1. Connectez-vous à l' Console de gestion AWS aide de vos informations d'identification d'utilisateur root (en particulier, l'adresse e-mail et le mot de passe que vous avez utilisés pour créer votre AWS compte).
2. Dans la barre de navigation, sélectionnez le nom de votre compte, puis Compte.
3. Faites défiler la page jusqu'à la section Accès des utilisateurs et des rôles IAM aux données de facturation, puis sélectionnez Modifier.
4. Cochez la case Activate IAM Access (Activer l'accès IAM) pour activer l'accès aux pages de la console de Gestion de la facturation et des coûts.
5. Choisissez Mettre à jour.

   La page affiche le message L' user/role accès IAM aux informations de facturation est activé.

   > **Important**
   >
   > L'activation de l'accès IAM n'accorde pas aux utilisateurs et aux rôles IAM les autorisations nécessaires pour accéder à ces pages de la console de facturation et gestion des coûts. Vous devez également associer les politiques d'identité requises aux rôles IAM pour accorder l'accès à la console de facturation. Les rôles fournissent des informations d'identification temporaires que les utilisateurs peuvent utiliser en cas de besoin.

6. Utilisez le Console de gestion AWS pour créer un rôle qu'un utilisateur peut assumer pour accéder à la console de facturation.
7. Sur la page Ajouter des autorisations du rôle, ajoutez des autorisations à la liste et consultez les détails des ressources de facturation de votre Compte AWS.

La politique AWS gérée Billing autorise les utilisateurs à consulter et à modifier la console de gestion de la facturation et des coûts. Cela inclut la consultation de l'utilisation du compte, ainsi que la modification des budgets et des modes de paiement. Pour d’autres exemples de politiques que vous pouvez associer aux rôles IAM afin de contrôler l’accès aux informations de facturation de votre compte, consultez Exemples de politiques de facturation AWS dans le Guide de l’utilisateur de Facturation et gestion des coûts.

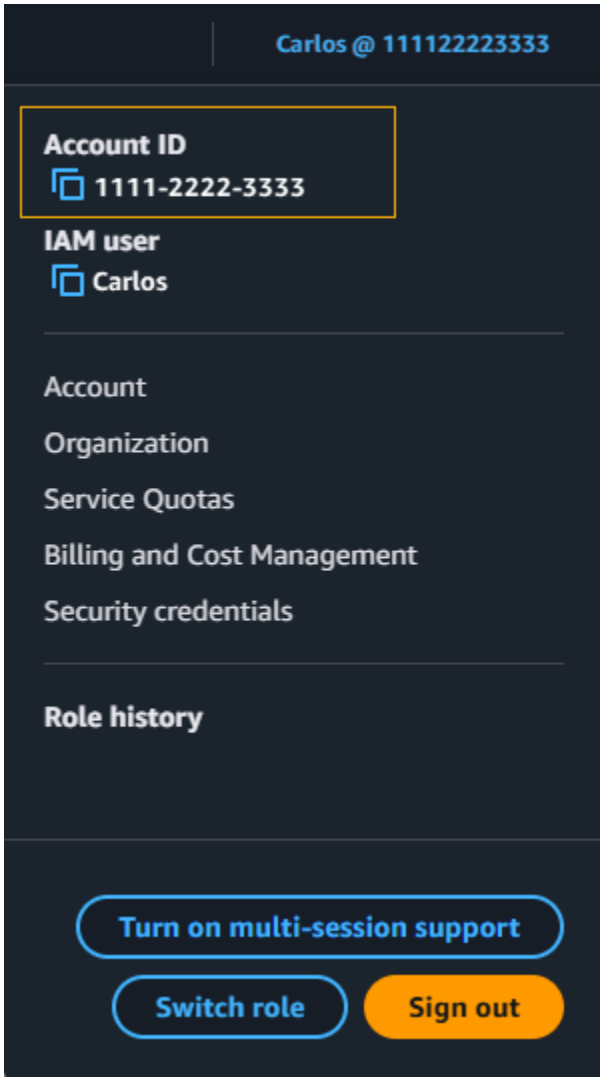
Visualisation de votre Compte AWS ID

Si vous êtes connecté à la console, vous pouvez consulter l’ID de compte de votre Compte AWS en utilisant les méthodes suivantes.

Pour consulter votre Compte AWS ID

Console

L'identifiant du AWS compte s'affiche lorsque vous accédez au tableau de bord IAM dans la Compte AWS section. Vous pouvez également consulter votre ID de compte dans la barre de navigation en haut à droite. Choisissez votre nom d’utilisateur, et l’ID du compte s’affiche au-dessus de votre nom d’utilisateur.



AWS CLI

Utilisez la commande suivante pour afficher votre ID utilisateur, votre ID de compte et votre ARN utilisateur :

- aws sts get-caller-identity

API

Utilisez l'API suivante pour afficher votre ID utilisateur, votre ID de compte et votre ARN utilisateur :

- GetCallerIdentity

Utiliser un alias pour votre Compte AWS ID

L'ID de compte est un numéro de 12 chiffres qui identifie votre compte de manière unique. Par défaut, les utilisateurs IAM du compte se connectent à l'aide d'une URL Web qui inclut l'ID du compte. S'ils ne disposent pas de l'URL, ils peuvent fournir l'identifiant du compte sur la page de AWS connexion lors de leur connexion.

Par défaut, l'URL de votre page de connexion utilise le format suivant.

```
https://Your_Account_ID.signin.aws.amazon.com/console/
```

De nombreuses personnes trouvent que les mots sont plus faciles à retenir que les chiffres. La création d'un alias pour votre ID de compte peut aider vos utilisateurs IAM à se connecter plus facilement.

Si vous créez un Compte AWS alias pour votre Compte AWS identifiant, l'URL de votre page de connexion ressemble à l'exemple suivant.

```
https://Your_Account_Alias.signin.aws.amazon.com/console/
```

Considérations à prendre en compte avant la création d'un alias de compte

- Vous ne Compte AWS pouvez avoir qu'un seul alias. Si vous créez un nouvel alias pour votre AWS compte, le nouvel alias remplace l'alias précédent et l'URL contenant l'alias précédent cesse de fonctionner.
- L'alias du compte ne doit contenir que des chiffres, des lettres minuscules et des traits d'union. Pour plus d'informations sur les restrictions applicables aux entités de AWS compte, consultezIAM et AWS STS quotas.
- L'alias du compte doit être unique pour tous les produits Amazon Web Services dans une partition de réseau donnée.

  Une partition est un groupe de régions AWS . Chaque AWS compte est limité à une partition.

  Les partitions prises en charge sont les suivantes :

  - `aws`- AWS Régions
  - `aws-cn` - Régions chinoises
  - `aws-us-gov` - AWS GovCloud (US) Régions

> **Note**
>
> Les alias de compte ne sont pas des secrets et ils apparaîtront dans l'URL de votre page de connexion publique. N'incluez aucune information sensible dans l'alias de votre compte. L'URL d'origine contenant votre Compte AWS identifiant reste active et peut être utilisée une fois que vous avez créé votre Compte AWS alias.

Pour obtenir des instructions sur la configuration ou la suppression d'un alias, consultez Création d'un alias de compte etSuppression d'un alias de compte.

Création d'un alias de compte

Pour effectuer les étapes suivantes, vous devez au moins disposer des autorisations IAM suivantes :

- `iam:ListAccountAliases`
- `iam:CreateAccountAlias`

Pour créer un Compte AWS alias

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, sélectionnez Dashboard (Tableau de bord).
3. Dans la section AWS Compte, près de Alias du compte, choisissez Créer. Si un alias existe déjà, sélectionnez Edit (Modifier).
4. Dans la boîte de dialogue, saisissez le nom que vous souhaitez utiliser pour l'alias, puis sélectionnez Enregistrer les modifications.

AWS CLI

Exécutez la commande suivante :

- `aws iam create-account-alias`

API

Pour créer un alias pour votre URL de la page de connexion de l' Console de gestion AWS , appelez l'opération suivante :

- `CreateAccountAlias`

Suppression d'un alias de compte

Pour effectuer les étapes suivantes, vous devez au moins disposer des autorisations IAM suivantes :

- `iam:ListAccountAliases`
- `iam:DeleteAccountAlias`

Pour supprimer un alias de compte

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, sélectionnez Dashboard (Tableau de bord).
3. Dans la section AWS Compte, près de l'Alias du compte, choisissez Supprimer.

AWS CLI

Pour supprimer un alias d' Compte AWS ID, exécutez la commande suivante :

- `aws iam delete-account-alias`

Pour confirmer que l'alias du compte est supprimé, essayez d'afficher l'alias de votre Compte AWS identifiant en exécutant la commande suivante :

- `aws iam list-account-aliases`

API

Pour supprimer un alias d' Compte AWS ID, effectuez l'opération suivante :

- DeleteAccountAlias

Pour confirmer que l'alias du compte est supprimé, essayez d'afficher l'alias de votre Compte AWS identifiant en effectuant l'opération suivante :

- ListAccountAliases

> **Note**
>
> Après avoir supprimé l'alias de votre compte, la seule URL de connexion pour votre compte est basée sur l'ID de votre compte. Toute tentative de connexion à l'URL alias échouera et ne sera pas redirigée.

Planifiez l'accès à votre AWS compte

Lors de la configuration AWS, planifiez la manière dont vous souhaitez que les utilisateurs accèdent à votre AWS compte et à vos ressources. Une solution de gestion des identités bien conçue assure la sécurité de votre compte.

Sources d'identité

Conformément aux meilleures pratiques IAM, nous recommandons aux utilisateurs humains et aux charges de travail d'utiliser des informations d'identification temporaires pour accéder à vos AWS ressources. Vous accordez des informations d'identification temporaires via les rôles IAM. Aussi bien les utilisateurs fédérés dans IAM que les utilisateurs d'IAM Identity Center (qu'ils soient fédérés ou créés dans l'annuaire d'IAM Identity Center) utilisent les rôles IAM pour accéder aux ressources.

Avant de commencer à utiliser AWS, planifiez comment configurer vos identités en utilisant l'une des approches suivantes :

- Activation d'IAM Identity Center avec IAM Identity Center AWS Organizations et ajout d'utilisateurs directement dans l'annuaire de l'organisation.

  Pour obtenir des instructions sur l'ajout d'utilisateurs directement au répertoire organisationnel d'IAM Identity Center, voir Ajouter des utilisateurs
- fédérant votre fournisseur d'identité externe existant soit avec IAM Identity Center, soit avec IAM.

Pour obtenir des instructions sur la fédération d'un fournisseur d'identité externe à l'annuaire organisationnel IAM Identity Center, consultez le didacticiel de démarrage approprié.

Gestion des accès

Identifiez les AWS ressources et les services dont vos utilisateurs ont besoin. Définissez ensuite les autorisations et les politiques d'accès pour chaque utilisateur, groupe ou rôle.

- Si vous utilisez IAM Identity Center, un fournisseur d'identité IAM, des rôles IAM et des politiques d'autorisations sont créés dans chaque AWS compte de votre organisation. Ces rôles et autorisations correspondent à ceux que vous spécifiez lorsque vous attribuez des personnes ou des groupes à des applications ou à AWS des comptes.

  Pour plus d'informations, consultez Attribuer un accès utilisateur et Configurer l'accès par authentification unique à vos applications.
- Vous pouvez utiliser le gestionnaire d'accès aux comptes, une fonctionnalité IAM qui vous permet d'attribuer des rôles IAM existants que vous créez et gérez aux utilisateurs et aux groupes IAM Identity Center. Le gestionnaire d'accès aux comptes vous donne accès à l'ensemble des fonctionnalités des rôles IAM. Vous pouvez l'utiliser conjointement avec des ensembles d'autorisations ou seul.
- Si vous fédérez votre fournisseur d'identité directement avec IAM dans votre système Compte AWS, vous devez créer un rôle que vos utilisateurs pourront assumer. Le rôle nécessite une politique de confiance et une politique d'autorisations. La politique de confiance indique qui peut assumer ce rôle. La politique d'autorisations indique AWS les actions et les ressources auxquelles le rôle est autorisé ou refusé.

  Pour de plus amples informations, consultez Fournisseurs d'identité et fédération au sein AWS.

Pour obtenir des instructions sur l'ajout d'un niveau de sécurité supplémentaire pour la connexion, consultezUtiliser l'authentification multifactorielle avec vos identités.

Cas d'utilisation pour les utilisateurs IAM

Les utilisateurs IAM que vous créez dans votre compte Compte AWS possèdent des informations d'identification à long terme que vous gérez directement.

Lorsqu'il s'agit de gérer les accès AWS, les utilisateurs IAM ne sont généralement pas le meilleur choix. Il existe quelques raisons principales pour lesquelles vous devriez éviter de vous fier aux utilisateurs IAM dans la plupart de vos cas d’utilisation.

Tout d’abord, les utilisateurs IAM sont conçus pour des comptes individuels, de sorte qu’ils ne s’adaptent pas bien à la croissance de votre organisation. La gestion des autorisations et de la sécurité pour un grand nombre d’utilisateurs IAM peut rapidement devenir un défi.

Les utilisateurs IAM n'ont pas non plus la visibilité centralisée et les fonctionnalités d'audit dont bénéficient les autres solutions de gestion des AWS identités. Cela peut compromettre le maintien de la sécurité et de la conformité aux réglementations.

Enfin, mettre en œuvre des pratiques exemplaires en matière de sécurité, comme l’authentification multifactorielle, les politiques de mot de passe et la séparation des rôles, est beaucoup plus facile avec des approches de gestion des identités plus évolutives.

Au lieu de vous fier aux utilisateurs IAM, nous vous recommandons d'utiliser des solutions plus robustes telles que IAM Identity Center avec AWS Organizations des fournisseurs externes ou des identités fédérées provenant de fournisseurs externes. Ces options vous permettront d'améliorer le contrôle, la sécurité et l'efficacité opérationnelle au fur et à mesure de l'évolution de votre AWS environnement.

Nous vous recommandons de n’utiliser les utilisateurs IAM que pour les cas d’utilisation non pris en charge par les utilisateurs fédérés.

La liste suivante répertorie les cas d'utilisation spécifiques qui nécessitent des informations d'identification à long terme auprès des utilisateurs IAM. AWS Vous pouvez utiliser IAM pour créer ces utilisateurs IAM dans le cadre de votre AWS compte, et utiliser IAM pour gérer leurs autorisations.

- Accès d'urgence à votre AWS compte
- Charges de travail qui ne peuvent pas utiliser des rôles IAM
  - AWS CodeCommit accès
  - Accès à Amazon Keyspaces (pour Apache Cassandra)
- Third-party AWS clients
- AWS IAM Identity Center n'est pas disponible pour votre compte et vous n'avez aucun autre fournisseur d'identité

Création d'un utilisateur IAM pour l'accès d'urgence

Un utilisateur IAM est une identité au sein de vous Compte AWS qui possède des autorisations spécifiques pour une seule personne ou une seule application.

Disposer d'un utilisateur IAM pour l'accès d'urgence est l'une des raisons recommandées pour créer un utilisateur IAM afin de pouvoir accéder à votre compte Compte AWS si votre fournisseur d'identité n'est pas accessible.

> **Note**
>
> Une bonne pratique en matière de sécurité consiste à fournir un accès à vos ressources par le biais de la fédération d'identité plutôt que de créer des utilisateurs IAM. Pour en savoir plus sur les situations spécifiques dans lesquelles un utilisateur IAM est nécessaire, veuillez consulter Quand créer un utilisateur IAM (au lieu d'un rôle).

Pour créer un utilisateur IAM pour l'accès d'urgence

> **Autorisations minimales**
>
> Pour effectuer les étapes suivantes, vous devez au moins disposer des autorisations IAM suivantes :
>
> - `access-analyzer:ValidatePolicy`
> - `iam:AddUserToGroup`
> - `iam:AttachGroupPolicy`
> - `iam:CreateGroup`
> - `iam:CreateLoginProfile`
> - `iam:CreateUser`
> - `iam:GetAccountPasswordPolicy`
> - `iam:GetLoginProfile`
> - `iam:GetUser`
> - `iam:ListAttachedGroupPolicies`
> - `iam:ListAttachedUserPolicies`
> - `iam:ListGroupPolicies`

- `iam:ListGroups`
- `iam:ListGroupsForUser`
- `iam:ListPolicies`
- `iam:ListUserPolicies`
- `iam:ListUsers`

Console

1. Suivez la procédure de connexion adaptée à votre type d'utilisateur, comme décrit dans la rubrique Comment se connecter AWS dans le Guide de l'AWS Sign-In utilisateur.
2. Sur la page d'accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le volet de navigation, sélectionnez Utilisateurs, puis Créer un utilisateur.

> **Note**
>
> Si IAM Identity Center est activé, un rappel Console de gestion AWS s'affiche pour vous rappeler qu'il est préférable de gérer l'accès des utilisateurs dans IAM Identity Center. Dans le cadre de cette procédure, l'utilisateur IAM que vous créez est spécifiquement destiné à n'être utilisé que lorsque votre fournisseur d'identité n'est pas disponible.

4. Sur la page Spécifier les détails de l'utilisateur, sous Détails de l'utilisateur, dans Nom d'utilisateur, entrez le nom du nouvel utilisateur. Il s'agit de son nom de connexion pour AWS. Pour cet exemple, entrez **`EmergencyAccess`**.

> **Note**
>
> Les noms d'utilisateur peuvent combiner jusqu'à 64 lettres, chiffres et caractères suivants : plus (+), égal (=), virgule (,), point (.), arobase (@), trait de soulignement (_) et tiret (-). Les noms doivent être uniques au sein d'un compte. Ils ne sont pas distingués au cas par cas. Par exemple, vous ne pouvez pas créer deux utilisateurs nommés TESTUSER et testuser. Lorsqu'un nom d'utilisateur est utilisé dans une politique ou dans le cadre d'un ARN, il est sensible à la casse. Lorsqu'un nom

d'utilisateur apparaît aux clients dans la console, par exemple lors du processus de connexion, il n'est pas sensible à la casse.

5. Cochez la case située en regard de Fournir un accès utilisateur à l' Console de gestion AWS– facultatif, puis sélectionnez Je souhaite créer un utilisateur IAM.
6. Sous Mot de passe de la console, sélectionnez Mot de passe généré automatiquement.
7. Décochez la case située en regard de l'option L'utilisateur doit générer un nouveau mot de passe à la prochaine connexion (recommandé). Comme cet utilisateur IAM est destiné à un accès d'urgence, un administrateur de confiance conserve le mot de passe et ne le fournit qu'en cas de besoin.
8. Sur la page Définir les autorisations, sous Options d'autorisations, sélectionnez Ajouter un utilisateur au groupe. Ensuite, sous Groupes d'utilisateurs, sélectionnez Créer un groupe.
9. Sur la page Créer un groupe d'utilisateurs, saisissez **`EmergencyAccessGroup`** dans le champ Nom du groupe d'utilisateurs. Ensuite, sous Politiques d'autorisations, sélectionnez AdministratorAccess.
10. Choisissez Créer un groupe d'utilisateurs pour revenir à la page Définir les autorisations.
11. Sous Groupes d'utilisateurs, sélectionnez le nom du groupe **`EmergencyAccessGroup`** précédemment créé.
12. Sélectionnez Suivant pour accéder à la page Vérifier et créer.
13. Sur la page Vérifier et créer, examinez la liste des appartenances aux groupes à attribuer au nouvel utilisateur. Une fois que vous êtes prêt à continuer, sélectionnez Créer l'utilisateur.
14. Sur la page Récupérer le mot de passe, sélectionnez Télécharger le fichier .csv pour enregistrer le fichier .csv contenant les informations d'identification de l'utilisateur (URL de connexion, nom d'utilisateur et mot de passe).
15. Enregistrez ce fichier pour l'utiliser si vous devez vous connecter à IAM et que vous n'avez pas accès à votre fournisseur d'identité.

Le nouvel utilisateur IAM apparaît dans la liste Utilisateurs. Cliquez sur le lien Nom d'utilisateur pour afficher les détails de l'utilisateur.

AWS CLI

1. Créez un utilisateur nommé **`EmergencyAccess`**.

   - aws iam create-user

```
aws iam create-user \
   --user-name EmergencyAccess
```

2. (Facultatif) Donnez à utilisateur l'accès à l'interface Console de gestion AWS. Un mot de passe est requis. Pour créer un mot de passe pour un utilisateur IAM, vous pouvez utiliser le paramètre `--cli-input-json` pour transmettre un fichier JSON contenant le mot de passe. Vous devez également fournir à l'utilisateur l'URL de la page de connexion de votre compte.

   - aws iam create-login-profile

   ```
   aws iam create-login-profile \
      --generate-cli-skeleton > create-login-profile.json
   ```

   - Ouvrez le fichier `create-login-profile.json` dans un éditeur de texte et saisissez un mot de passe conforme à votre politique en matière de mots de passe, puis enregistrez le fichier. Par exemple :

   ```
   {
    "UserName": "EmergencyAccess",
    "Password": "Ex@3dRA0djs",
    "PasswordResetRequired": false
   }
   ```

   - Réutilisez la commande `aws iam create-login-profile` en passant le paramètre `--cli-input-json` pour spécifier votre fichier JSON.

   ```
   aws iam create-login-profile \
      --cli-input-json file://create-login-profile.json
   ```

> **Note**
>
> Si le mot de passe que vous avez fourni dans le fichier JSON enfreint la politique de mot de passe de votre compte, vous recevrez une erreur `PasswordPolicyViolation`. Dans ce cas, consultez la politique de mot de passe de votre compte et mettez à jour le mot de passe dans le fichier JSON afin de respecter les exigences.

3. Créez le**`EmergencyAccessGroup`**, attachez la politique AWS gérée `AdministratorAccess` au groupe et ajoutez l'**`EmergencyAccess`**utilisateur au groupe.

> **Note**
>
> Une politique gérée par AWS est une politique autonome qui est créée et gérée par AWS. Chaque politique possède son propre Amazon Resource Name (ARN) dans lequel figure le nom de la politique. Par exemple, il `arn:aws:iam::aws:policy/IAMReadOnlyAccess` s'agit d'une politique AWS gérée. Pour plus d'informations sur les ARN, consultez ARN IAM. Pour obtenir la liste des politiques AWS gérées pour Services AWS, consultez la section Politiques AWS gérées.

- aws iam create-group

```
aws iam create-group \
   --group-name EmergencyAccessGroup
```

- aws iam attach-group-policy

```
aws iam attach-group-policy \
   --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \
   --group-name >EmergencyAccessGroup
```

- aws iam add-user-to-group

```
aws iam add-user-to-group \
```

```
    --user-name EmergencyAccess \
    --group-name EmergencyAccessGroup
```

- Exécutez la commande aws iam get-group afficher la liste de **EmergencyAccessGroup** et de ses membres.

```
aws iam get-group \
    --group-name EmergencyAccessGroup
```

Création d'un utilisateur IAM pour les charges de travail qui ne peuvent pas utiliser de rôles IAM

> **Important**
>
> En tant que meilleure pratique, nous vous recommandons de demander à vos utilisateurs humains d'utiliser des informations d'identification temporaires lors de l'accès AWS. Vous pouvez également gérer vos identités d'utilisateur, y compris votre utilisateur administratif, avec AWS IAM Identity Center. Nous vous recommandons d'utiliser IAM Identity Center pour gérer l'accès à vos comptes et les autorisations au sein de ceux-ci. Si vous utilisez un fournisseur d'identité externe, vous pouvez également configurer les autorisations d'accès pour les identités des utilisateurs dans IAM Identity Center.

Si votre cas d'utilisation nécessite des utilisateurs IAM disposant d'un accès programmatique et d'informations d'identification à long terme, nous vous recommandons d'établir des procédures de mise à jour des clés d'accès. Pour de plus amples informations, veuillez consulter Mise à jour des clés d'accès.

Pour effectuer certaines tâches de gestion des comptes et des services, vous devez vous connecter à l'aide des informations d'identification de l'utilisateur racine. Pour afficher les tâches qui nécessitent que vous vous connectiez en tant qu'utilisateur racine, consultez Tâches nécessitant les informations d'identification de l'utilisateur root.

Pour créer un utilisateur IAM pour les charges de travail qui ne peuvent pas utiliser de rôles IAM

Autorisations minimales

Pour effectuer les étapes suivantes, vous devez au moins disposer des autorisations IAM suivantes :

- `iam:AddUserToGroup`
- `iam:AttachGroupPolicy`
- `iam:CreateAccessKey`
- `iam:CreateGroup`
- `iam:CreateServiceSpecificCredential`
- `iam:CreateUser`
- `iam:GetAccessKeyLastUsed`
- `iam:GetAccountPasswordPolicy`
- `iam:GetAccountSummary`
- `iam:GetGroup`
- `iam:GetLoginProfile`
- `iam:GetPolicy`
- `iam:GetRole`
- `iam:GetUser`
- `iam:ListAccessKeys`
- `iam:ListAttachedGroupPolicies`
- `iam:ListAttachedUserPolicies`
- `iam:ListGroupPolicies`
- `iam:ListGroups`
- `iam:ListGroupsForUser`
- `iam:ListInstanceProfilesForRole`
- `iam:ListMFADevices`
- `iam:ListPolicies`
- `iam:ListRoles`
- `iam:ListRoleTags`

- `iam:ListSSHPublicKeys`
- `iam:ListServiceSpecificCredentials`
- `iam:ListSigningCertificates`
- `iam:ListUserPolicies`
- `iam:ListUserTags`
- `iam:ListUsers`
- `iam:UploadSSHPublicKey`
- `iam:UploadSigningCertificate`

Console

1. Suivez la procédure de connexion adaptée à votre type d'utilisateur, comme décrit dans la rubrique Comment se connecter AWS dans le Guide de l'AWS Sign-In utilisateur.
2. Sur la page d'accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le volet de navigation, choisissez Utilisateurs, puis sélectionnez Créer un utilisateur.
4. Sur la page Spécification de détails de l'utilisateur, procédez comme suit :
   a. Pour Nom d'utilisateur, saisissez ***`WorkloadName`***. Remplacez ***`WorkloadName`*** par le nom de la charge de travail qui utilisera le compte.
   b. Choisissez Suivant.
5. (Facultatif) Sur la page Définir les autorisations, procédez comme suit :
   a. Choisissez ajouter un utilisateur au groupe.
   b. Choisissez Créer un groupe.
   c. Dans la boîte de dialogue Créer un groupe d'utilisateurs, pour Nom du groupe d'utilisateurs, saisissez un nom qui reflète l'utilisation des charges de travail du groupe. Pour cet exemple, utilisez le nom **`Automation`**.
   d. Sous Politiques d'autorisations, cochez la case correspondant à la politique PowerUserAccess gérée.

> **Tip**
>
> Saisissez Power dans le champ de recherche des Politiques d'autorisations pour trouver rapidement la politique gérée.

e. Choisissez Créer un groupe d'utilisateurs.

f. De retour sur la page avec la liste des groupes IAM, cochez la case de votre nouveau groupe d'utilisateurs. Sélectionnez Refresh (Actualiser) si vous ne voyez pas le nouveau groupe d'utilisateurs dans la liste.

g. Choisissez Suivant.

6. (Facultatif) Dans la section Balises, ajoutez des métadonnées à l'utilisateur en associant les balises sous forme de paires clé-valeur. Pour de plus amples informations, veuillez consulter Tags pour Gestion des identités et des accès AWS resources.

7. Vérifiez l'appartenance du nouvel utilisateur à un groupe d'utilisateurs. Une fois que vous êtes prêt à continuer, choisissez Create user (Créer un utilisateur).

8. Une notification de statut apparaît pour vous informer que l'utilisateur a été créé avec succès. Sélectionnez Afficher l'utilisateur pour accéder à la page des détails de l'utilisateur.

9. Sélectionnez l'onglet Informations d'identification de sécurité. Créez ensuite les informations d'identification requises pour la charge de travail.

   - Clés d'accès : sélectionnez Créer une clé d'accès pour générer et télécharger des clés d'accès pour l'utilisateur.

     > **Important**
     >
     > Il s'agit de votre seule opportunité de consulter ou de télécharger les clés d'accès secrètes, et vous devez fournir ces informations à vos utilisateurs avant qu'ils puissent utiliser l' AWS API. Enregistrez les nouveaux ID de clé d'accès et clé d'accès secrète de l'utilisateur dans un endroit sûr et sécurisé. Vous ne pourrez plus accéder aux clés d'accès secrètes après cette étape.

   - Clés publiques SSH pour AWS CodeCommit : sélectionnez Télécharger la clé publique SSH pour télécharger une clé publique SSH afin que l'utilisateur puisse communiquer avec les CodeCommit référentiels via SSH.

- Informations d'identification Git HTTPS pour AWS CodeCommit : sélectionnez Générer des informations d'identification pour générer un ensemble unique d'informations d'identification utilisateur à utiliser avec les référentiels Git. Sélectionnez Télécharger les informations d'identification pour enregistrer le nom d'utilisateur et le mot de passe dans un fichier .csv. C'est la seule fois où cette information est disponible. Si vous oubliez ou perdez le mot de passe, vous devrez le réinitialiser.
- Informations d'identification pour Amazon Keyspaces (pour Apache Cassandra) : sélectionnez Générer des informations d'identification pour générer des informations d'identification utilisateur spécifiques au service à utiliser avec Amazon Keyspaces. Sélectionnez Télécharger les informations d'identification pour enregistrer le nom d'utilisateur et le mot de passe dans un fichier .csv. C'est la seule fois où cette information est disponible. Si vous oubliez ou perdez le mot de passe, vous devrez le réinitialiser.

  > **Important**
  >
  > Service-specific les informations d'identification sont des informations d'identification à long terme associées à un utilisateur IAM spécifique et ne peuvent être utilisées que pour le service pour lequel elles ont été créées. Pour autoriser les rôles IAM ou les identités fédérées à accéder à toutes vos AWS ressources à l'aide d'informations d'identification temporaires, utilisez l' AWS authentification avec le plug-in d'authentification Sigv4 pour Amazon Keyspaces. Pour de plus d'informations, consultez Utilisation d'informations d'identification temporaires pour se connecter à Amazon Keyspaces à l'aide d'un rôle IAM et du plug-in SigV4 dans le Guide du développeur Amazon Keyspaces (pour Apache Cassandra).

- X.509 Certificats de signature : sélectionnez Créer un X.509 certificat si vous devez effectuer des SOAP-protocol demandes sécurisées et que vous vous trouvez dans une région qui n'est pas prise en charge par AWS Certificate Manager. ACM est l'outil préféré pour mettre en service, gérer et déployer vos certificats de serveur. Pour plus d'informations sur l'utilisation d'ACM, consultez le Guide de l'utilisateur AWS Certificate Manager .

Vous avez créé un utilisateur avec accès programmatique et l'avez configuré avec la fonction PowerUserAccess job. La politique d'autorisations de cet utilisateur accorde un accès complet à tous les services, à l'exception d'IAM et AWS Organizations.

Vous pouvez utiliser ce même processus pour donner à des charges de travail supplémentaires un accès programmatique à vos Compte AWS ressources, si les charges de travail ne sont pas en mesure d'assumer des rôles IAM. Cette procédure a utilisé la politique PowerUserAccess gérée pour attribuer des autorisations. Pour respecter les pratiques exemplaires du moindre privilège, envisagez d'utiliser une politique plus restrictive ou de créer une politique personnalisée qui limite l'accès aux seules ressources requises par le programme. Pour en savoir plus sur l'utilisation de politiques qui limitent les autorisations des utilisateurs à AWS des ressources spécifiques, consultez Gestion des accès pour AWS resources etExemples de politiques basées sur l'identité IAM. Pour ajouter des utilisateurs supplémentaires au groupe d'utilisateurs une fois celui-ci créé, veuillez consulter Modification des utilisateurs dans les groupes IAM.

AWS CLI

1. Créez un utilisateur nommé **`Automation`**.

   - aws iam create-user

   ```
   aws iam create-user \
       --user-name Automation
   ```

2. Créez un groupe d'utilisateurs IAM nommé**`AutomationGroup`**, associez la politique AWS gérée `PowerUserAccess` au groupe, puis ajoutez l'**`Automation`**utilisateur au groupe.

   > **Note**
   >
   > Une politique gérée par AWS est une politique autonome qui est créée et gérée par AWS. Chaque politique possède son propre Amazon Resource Name (ARN) dans lequel figure le nom de la politique. Par exemple, il `arn:aws:iam::aws:policy/IAMReadOnlyAccess` s'agit d'une politique AWS gérée. Pour plus d'informations sur les ARN, consultez ARN IAM. Pour obtenir la liste des politiques AWS gérées pour Services AWS, consultez la section Politiques AWS gérées.

   - aws iam create-group

   ```
   aws iam create-group \
   ```

```
                    --group-name AutomationGroup
```

- aws iam attach-group-policy

```
aws iam attach-group-policy \
    --policy-arn arn:aws:iam::aws:policy/PowerUserAccess \
    --group-name AutomationGroup
```

- aws iam add-user-to-group

```
aws iam add-user-to-group \
    --user-name Automation \
    --group-name AutomationGroup
```

- Exécutez la commande aws iam get-group afficher la liste de **AutomationGroup** et de ses membres.

```
aws iam get-group \
    --group-name AutomationGroup
```

3. Créez les informations d'identification de sécurité requises pour la charge de travail.

   - Créer des clés d'accès pour les tests–aws iam create-access-key

```
aws iam create-access-key \
    --user-name Automation
```

   La sortie de cette commande affiche la clé d'accès secrète et l'ID de clé d'accès. Enregistrez et stockez ces informations dans un emplacement sûr. Si ces informations d'identification sont perdues, elles ne peuvent pas être récupérées et vous devez générer une nouvelle clé d'accès.

> **Important**
>
> Ces clés d'accès utilisateur IAM sont des informations d'identification à long terme qui présentent un risque de sécurité pour votre compte. Une fois les tests terminés, nous vous recommandons de supprimer ces clés d'accès. Si vous envisagez d'utiliser des clés d'accès dans certains scénarios, vérifiez si vous pouvez activer la MFA pour l'utilisateur IAM de votre charge de travail et utiliser aws sts get-session-token pour obtenir des informations d'identification temporaires pour la session au lieu d'utiliser des clés d'accès IAM.

- Télécharger des clés publiques SSH pour AWS CodeCommit — aws iam upload-ssh-public-key

  L'exemple suivant suppose que vos clés publiques SSH sont stockées dans le fichier `sshkey.pub`.

  ```
  aws upload-ssh-public-key \
      --user-name Automation \
      --ssh-public-key-body file://sshkey.pub
  ```

- Télécharger et X.509 signer un certificat — aws iam upload-signing-certificate

  Téléchargez un X.509 certificat si vous devez effectuer des SOAP-protocol demandes sécurisées et que vous vous trouvez dans une région qui n'est pas prise en charge par AWS Certificate Manager. ACM est l'outil préféré pour mettre en service, gérer et déployer vos certificats de serveur. Pour plus d'informations sur l'utilisation d'ACM, consultez le Guide de l'utilisateur AWS Certificate Manager .

  L'exemple suivant suppose que votre certificat de X.509 signature est stocké dans le fichier`certificate.pem`.

  ```
  aws iam upload-signing-certificate \
  --user-name Automation \
  --certificate-body file://certificate.pem
  ```

Vous pouvez utiliser ce même processus pour donner à des charges de travail supplémentaires un accès programmatique à vos Compte AWS ressources, si les charges de travail ne sont pas en mesure d'assumer des rôles IAM. Cette procédure a utilisé la politique PowerUserAccess gérée pour attribuer des autorisations. Pour respecter les pratiques exemplaires du moindre privilège, envisagez d'utiliser une politique plus restrictive ou de créer une politique personnalisée qui limite l'accès aux seules ressources requises par le programme. Pour en savoir plus sur l'utilisation de politiques qui limitent les autorisations des utilisateurs à AWS des ressources spécifiques, consultez Gestion des accès pour AWS resources etExemples de politiques basées sur l'identité IAM. Pour ajouter des utilisateurs supplémentaires au groupe d'utilisateurs une fois celui-ci créé, veuillez consulter Modification des utilisateurs dans les groupes IAM.

Utiliser l'authentification multifactorielle avec vos identités

L'utilisation de l'authentification multifactorielle (MFA) avec vos identités est une autre bonne pratique en matière d'IAM. La MFA est une couche de sécurité supplémentaire qui exige des utilisateurs qu'ils fournissent des facteurs d'authentification supplémentaires après avoir fourni leur nom d'utilisateur et leur mot de passe pour vérifier leur identité. Elle renforce considérablement la sécurité en rendant l'accès non autorisé beaucoup plus difficile pour les attaquants, même si le mot de passe d'un utilisateur est compromis. La MFA est largement adoptée comme une bonne pratique pour sécuriser l'accès aux comptes en ligne, aux services cloud et à d'autres ressources sensibles. AWS prend en charge le MFA pour l'utilisateur root, les utilisateurs IAM, les utilisateurs d'IAM Identity Center, le Builder ID et les utilisateurs fédérés. Pour plus de sécurité, vous pouvez créer des politiques qui exigent que l'authentification MFA soit configurée avant d'autoriser un utilisateur à accéder aux ressources ou à effectuer des actions spécifiques et associer ces politiques à vos rôles IAM. IAM Identity Center est préconfiguré avec la MFA activée par défaut, de sorte que tous les utilisateurs d'IAM Identity Center doivent se connecter avec la MFA en plus de leur nom d'utilisateur et de leur mot de passe.

> **Note**
>
> Tous les Compte AWS types (comptes autonomes, de gestion et comptes membres) nécessitent que l'authentification MFA soit configurée pour leur utilisateur root. Les utilisateurs doivent enregistrer le MFA dans les 35 jours suivant leur première tentative de connexion pour accéder au si le Console de gestion AWS MFA n'est pas déjà activé.

Pour plus d'informations, consultez Configurer la MFA dans IAM Identity Center et AWS Multi-factor authentification dans IAM.

Préparation des autorisations de moindre privilège

L'utilisation d'autorisations de moindre privilège est une recommandation de bonne pratique IAM. Le concept des autorisations de moindre privilège consiste à accorder aux utilisateurs les autorisations nécessaires à l'exécution d'une tâche et aucune autorisation supplémentaire. Lors des préparatifs, réfléchissez à la manière dont vous allez prendre en charge les autorisations de moindre privilège. L'utilisateur racine, l'utilisateur administratif et l'utilisateur IAM d'accès d'urgence disposent d'autorisations puissantes qui ne sont pas nécessaires pour les tâches quotidiennes. Pendant que vous découvrez AWS et testez différents services, nous vous recommandons de créer au moins un utilisateur supplémentaire dans IAM Identity Center avec des autorisations moindres que vous pouvez utiliser dans différents scénarios. Vous pouvez utiliser les politiques IAM pour définir les actions qui peuvent être entreprises sur des ressources données dans des conditions spécifiques, puis vous connecter à ces ressources avec le compte doté de privilèges moindres.

Si vous utilisez IAM Identity Center, pensez à utiliser des jeux d'autorisations IAM Identity Center pour commencer. Pour en savoir plus, veuillez consulter la rubrique Create a permission set dans le Guide de l'utilisateur IAM Identity Center.

Si vous n'utilisez pas IAM Identity Center, utilisez les rôles IAM pour définir les autorisations des différentes entités IAM. Pour en savoir plus, veuillez consulter la section Création d'un rôle IAM.

Les rôles IAM et les ensembles d'autorisations IAM Identity Center peuvent utiliser des politiques AWS gérées en fonction des fonctions du poste. Pour plus de détails sur les autorisations accordées par ces politiques, consultez AWS politiques gérées pour les fonctions professionnelles.

> **Important**
>
> N'oubliez pas que les politiques AWS gérées peuvent ne pas accorder les autorisations de moindre privilège pour vos cas d'utilisation spécifiques, car elles sont disponibles pour tous AWS les clients. Au terme de la configuration, nous vous recommandons d'utiliser IAM Access Analyzer pour générer des stratégies de moindre privilège basées sur l'activité d'accès consignée dans AWS CloudTrail. Pour plus d'informations sur la génération de politiques, consultez IAM Access Analyzer policy generation.

Lorsque vous commencez, nous vous recommandons d'utiliser des politiques AWS gérées pour accorder des autorisations. Après une période d'activité prédéfinie (90 jours, par exemple), vous pouvez examiner les services auxquels les personnes et les charges de travail ont accédé. Vous pouvez ensuite créer une nouvelle politique gérée par le client avec des autorisations réduites pour remplacer la politique AWS gérée. La nouvelle politique ne devrait inclure que les services auxquels il a été fait appel au cours de la période examinée. Mettez à jour vos autorisations pour supprimer la politique AWS gérée et y joindre la nouvelle politique gérée par le client que vous avez créée.

Pour obtenir des instructions sur l'affinement des autorisations, consultezRévision des dernières informations consultées pour votre AWS compte. Après avoir identifié les services non utilisés, consultezGénération d'une politique basée sur l'activité d'accès.

Révision des dernières informations consultées pour votre AWS compte

Vous pouvez consulter les informations du dernier accès au service pour IAM à l'aide de la console IAM ou AWS de l' AWS CLI API. Pour connaître les informations importantes sur les données, les autorisations requises, la résolution de problèmes et les régions prises en charge, veuillez consulter Affiner les autorisations en AWS utilisant les dernières informations consultées.

Vous pouvez consulter les informations relatives aux types de ressources suivants dans IAM. Dans chaque cas, les informations incluent les services autorisés pour la période donnée :

- Utilisateur IAM : affichez la dernière tentative d'accès de l'utilisateur à chaque service autorisé.
- Groupe IAM : affichez les informations relatives à la dernière tentative d'accès d'un membre du groupe IAM à chaque service autorisé. Ce rapport inclut également le nombre total de membres qui ont tenté un accès.
- Rôle IAM : affichez la dernière tentative d'accès à chaque service autorisé par une personne ayant utilisé ce rôle.
- Policy (Politique) : consultez les informations sur la dernière fois où un utilisateur ou un rôle a tenté d'accéder à chaque service autorisé. Ce rapport inclut également le nombre total d'entités qui ont tenté un accès.

> **Note**
>
> Avant de consulter les informations d'accès d'une ressource IAM, vous devez bien comprendre la période couverte par le rapport, les entités rapportées et les types de politique

évalués pour vos informations. Pour en savoir plus, consultez the section called "Choses à savoir sur les dernières informations consultées".

Pour de plus amples informations sur les dernières informations consultées, veuillez consulter Affiner les autorisations en AWS utilisant les dernières informations consultées.

Pour consulter les dernières informations consultées pour un Compte AWS

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d'accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), Users (Utilisateurs), Roles (Rôles) ou Policies (Politiques).
4. Choisissez un nom d'utilisateur, de groupe d'utilisateurs, de rôle ou de politique pour ouvrir la page Récapitulatif et sélectionnez l'onglet Dernier accès. Affichez les informations suivantes, en fonction de la ressource que vous avez choisie :

   - User group (Groupe d'utilisateurs) : affiche la liste des services auxquels les membres du groupe d'utilisateurs peuvent accéder. Vous pouvez également afficher la dernière fois qu'un membre a accédé au service, quelles politiques de groupe d'utilisateurs il a utilisées et quel membre du groupe d'utilisateurs a effectué la demande. Choisissez le nom de la politique pour savoir s'il s'agit d'une politique gérée ou d'une politique de groupe d'utilisateurs en ligne. Choisissez le nom du membre du groupe pour afficher tous les membres du groupe d'utilisateurs et quand ils ont consulté pour la dernière fois le service.
   - User (Utilisateur) : affiche la liste des services auxquels l'utilisateur peut accéder. Vous pouvez également afficher la date à laquelle ils ont accédé au service pour la dernière fois et les politiques qui sont actuellement associées à l'utilisateur. Choisissez le nom de la politique pour savoir s'il s'agit d'une politique gérée, d'une politique d'utilisateur en ligne ou d'une politique en ligne pour le groupe.
   - Role (Rôle) : affichez la liste des services auxquels le rôle peut accéder, à quel moment le rôle a accédé au service pour la dernière fois et quelles politiques ont été utilisées. Choisissez le nom de la politique pour savoir s'il s'agit d'une politique gérée ou d'une politique de rôle en ligne.

- Policy (Politique) : affiche la liste des services avec des actions autorisées dans la politique. Vous pouvez également afficher la dernière fois où la politique a été utilisée pour accéder au service et quelle entité (utilisateur ou rôle) l'a utilisé. La date Dernier accès inclut également le moment où l'accès est accordé à cette politique par l'intermédiaire d'une autre politique. Choisissez le nom de l'entité pour connaître les entités ayant cette politique attachée et quand elles ont consulté pour la dernière fois le service.

5. Dans la colonne Service du tableau, choisissez le nom de l'un des services qui inclut les dernières informations consultées relatives à une action pour afficher une liste des actions de gestion auxquelles les entités IAM ont tenté d'accéder. Vous pouvez afficher l' Région AWS et un horodatage indiquant la dernière fois que quelqu'un a tenté d'exécuter l'action.
6. La colonne Dernier accès s'affiche pour les services et les actions de gestion des services qui incluent les dernières informations consultées relatives à une action. Examinez les résultats possibles suivants qui sont renvoyés dans cette colonne. Ils varient selon qu'un service ou une action est autorisé, qu'il a été consulté et qu'il est suivi par AWS pour les dernières informations consultées.

   il y a <nombre de> jours

   Nombre de jours écoulés depuis que le service ou l'action a été utilisé pendant la période de suivi. La période de suivi des services s'étend sur les 400 derniers jours. La période de suivi des actions Amazon S3 a commencé le 12 avril 2020. La période de suivi des actions Amazon EC2, IAM et Lambda a commencé le 7 avril 2021. La période de suivi pour tous les autres services a débuté le 23 mai 2023. Pour en savoir plus sur les dates de début du suivi pour chacune d'entre elles Région AWS, consultezOù AWS suit les dernières informations consultées.

   Aucun accès pendant la période de suivi

   Le service ou l'action suivi n'a pas été utilisé par une entité au cours de la période de suivi.

   Vous pouvez disposez d'autorisations pour une action qui n'apparaît pas dans la liste. C'est le cas lorsque les informations de suivi de l'action ne sont pas actuellement prises en compte par AWS. Vous ne devez pas prendre de décisions sur les autorisations uniquement en raison de l'absence d'informations de suivi. Nous vous recommandons plutôt d'utiliser ces informations pour éclairer et fonder votre stratégie globale d'octroi du moindre privilège. Vérifiez vos politiques pour confirmer que le niveau d'accès est approprié.

AWS CLI

Vous pouvez utiliser le AWS CLI pour récupérer des informations sur la dernière fois qu'une ressource IAM de votre choix Compte AWS a été utilisée pour tenter d'accéder à des AWS services et à des actions Amazon S3, Amazon EC2, IAM et Lambda. Une ressource IAM peut être un utilisateur, un groupe d'utilisateurs, un rôle ou une politique.

- Générez un rapport pour les ressources IAM d'un Compte AWS. La demande doit inclure l'ARN de la ressource IAM (utilisateur, groupe d'utilisateurs, rôle ou politique) pour lequel vous voulez un rapport. Vous pouvez spécifier le niveau de granularité que vous souhaitez générer dans le rapport pour afficher les détails sur l'accès pour les services ou pour les services et les actions. La demande renvoie un `job-id` que vous pouvez ensuite utiliser dans les opérations `get-service-last-accessed-details-with-entities` et `get-service-last-accessed-details` pour surveiller l'état `job-status` jusqu'à ce que la tâche soit terminée.

  - aws iam -détails generate-service-last-accessed

  a. Récupérez les informations sur le rapport à l'aide du paramètre `job-id` de l'étape précédente.

     - aws iam -détails get-service-last-accessed

     Cette opération renvoie les informations suivantes, en fonction du type de ressource et du niveau de granularité que vous avez demandé dans l'opération `generate-service-last-accessed-details` :

     - User (Utilisateur) : renvoie une liste de services auxquels l'utilisateur spécifié peut accéder. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative de l'utilisateur, ainsi que l'ARN de l'utilisateur.
     - Groupe d'utilisateurs : renvoie la liste des services auxquels les membres du groupe d'utilisateurs spécifié peuvent accéder à l'aide des politiques attachées au groupe d'utilisateurs. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative effectuée par un membre du groupe d'utilisateurs. Il renvoie également l'ARN de cet utilisateur et le nombre total de membres du groupe d'utilisateurs qui ont tenté d'accéder au service. Utilisez cette

GetServiceLastAccessedDetailsWithEntitiesopération pour récupérer la liste de tous les membres.

- Role (Rôle) : renvoie la liste des services auxquels le rôle spécifié peut accéder. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative du rôle, ainsi que l'ARN du rôle.
- Policy (Politique) : renvoie la liste des services pour lesquels la politique spécifiée autorise l'accès. Pour chaque service, l'opération renvoie la date et l'heure auxquelles une entité (utilisateur ou rôle) a tenté pour la dernière fois d'accéder au service à l'aide de la politique. Elle renvoie également l'ARN de cette entité et le nombre total d'entités ayant tenté l'accès.

b. En savoir plus sur les entités qui ont utilisé les autorisations de groupe d'utilisateurs ou de politique lors de la tentative d'accès à un service spécifique. Cette opération renvoie la liste des entités avec l'ARN, l'ID, le nom, le chemin, le type (utilisateur ou rôle) de chaque entité, et la date à laquelle elles ont tenté pour la dernière fois d'accéder au service. Vous pouvez également utiliser cette opération pour les utilisateurs et les rôles, mais elle ne renvoie que les informations relatives à cette entité.

   - c'est moi get-service-last-accessed qui... details-with-entities

c. En savoir plus sur les politiques basées sur une identité qu'une identité (utilisateur, groupe d'utilisateurs ou rôle) a utilisées lors de la tentative d'accès à un service spécifique. Lorsque vous spécifiez une identité et un service, cette opération renvoie une liste de stratégies d'autorisation que l'identité peut utiliser pour accéder au service spécifié. Cette opération donne l'état actuel des stratégies et ne dépend pas du rapport généré. Il ne renvoie pas non plus d'autres types de politiques, tels que les politiques basées sur les ressources, les listes de contrôle d'accès, les AWS Organizations politiques, les limites d'autorisations IAM ou les politiques de session. Pour plus d'informations, consultez Types de politique ou Évaluation des politiques pour les demandes au sein d'un même compte.

   - aws iam -access list-policies-granting-service

API

Vous pouvez utiliser l' AWS API pour récupérer des informations sur la dernière fois qu'une ressource IAM a été utilisée pour tenter d'accéder à des AWS services et à des actions Amazon S3, Amazon EC2, IAM et Lambda. Une ressource IAM peut être un utilisateur, un groupe

d'utilisateurs, un rôle ou une politique. Vous pouvez spécifier le niveau de granularité à générer dans le rapport pour afficher les détails des services ou des services et des actions.

1. Générez un rapport. La demande doit inclure l'ARN de la ressource IAM (utilisateur, groupe d'utilisateurs, rôle ou politique) pour lequel vous voulez un rapport. Elle renvoie un `JobId` que vous pouvez ensuite utiliser dans les opérations `GetServiceLastAccessedDetailsWithEntities` et `GetServiceLastAccessedDetails` pour surveiller l'état `JobStatus` jusqu'à ce que la tâche soit terminée.

   - GenerateServiceLastAccessedDetails

2. Récupérez les informations sur le rapport à l'aide du paramètre `JobId` de l'étape précédente.

   - GetServiceLastAccessedDetails

   Cette opération renvoie les informations suivantes, en fonction du type de ressource et du niveau de granularité que vous avez demandé dans l'opération `GenerateServiceLastAccessedDetails` :

   - User (Utilisateur) : renvoie une liste de services auxquels l'utilisateur spécifié peut accéder. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative de l'utilisateur, ainsi que l'ARN de l'utilisateur.
   - Groupe d'utilisateurs : renvoie la liste des services auxquels les membres du groupe d'utilisateurs spécifié peuvent accéder à l'aide des politiques attachées au groupe d'utilisateurs. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative effectuée par un membre du groupe d'utilisateurs. Il renvoie également l'ARN de cet utilisateur et le nombre total de membres du groupe d'utilisateurs qui ont tenté d'accéder au service. Utilisez cette GetServiceLastAccessedDetailsWithEntitiesopération pour récupérer la liste de tous les membres.
   - Role (Rôle) : renvoie la liste des services auxquels le rôle spécifié peut accéder. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative du rôle, ainsi que l'ARN du rôle.
   - Policy (Politique) : renvoie la liste des services pour lesquels la politique spécifiée autorise l'accès. Pour chaque service, l'opération renvoie la date et l'heure auxquelles une entité (utilisateur ou rôle) a tenté pour la dernière fois d'accéder au service à l'aide de la politique. Elle renvoie également l'ARN de cette entité et le nombre total d'entités ayant tenté l'accès.

3. En savoir plus sur les entités qui ont utilisé les autorisations de groupe d'utilisateurs ou de politique lors de la tentative d'accès à un service spécifique. Cette opération renvoie la liste des entités avec l'ARN, l'ID, le nom, le chemin, le type (utilisateur ou rôle) de chaque entité, et la date à laquelle elles ont tenté pour la dernière fois d'accéder au service. Vous pouvez également utiliser cette opération pour les utilisateurs et les rôles, mais elle ne renvoie que les informations relatives à cette entité.

   - GetServiceLastAccessedDetailsWithEntities

4. En savoir plus sur les politiques basées sur une identité qu'une identité (utilisateur, groupe d'utilisateurs ou rôle) a utilisées lors de la tentative d'accès à un service spécifique. Lorsque vous spécifiez une identité et un service, cette opération renvoie une liste de stratégies d'autorisation que l'identité peut utiliser pour accéder au service spécifié. Cette opération donne l'état actuel des stratégies et ne dépend pas du rapport généré. Il ne renvoie pas non plus d'autres types de politiques, tels que les politiques basées sur les ressources, les listes de contrôle d'accès, les AWS Organizations politiques, les limites d'autorisations IAM ou les politiques de session. Pour plus d'informations, consultez Types de politique ou Évaluation des politiques pour les demandes au sein d'un même compte.

   - ListPoliciesGrantingServiceAccess

Génération d'une politique basée sur l'activité d'accès

Vous pouvez utiliser l'activité d'accès enregistrée AWS CloudTrail pour un utilisateur ou un rôle IAM pour qu'IAM Access Analyzer génère une politique gérée par le client afin de n'autoriser l'accès qu'aux services dont des utilisateurs et des rôles spécifiques ont besoin.

Lorsque IAM Access Analyzer génère une politique IAM, des informations sont renvoyées pour vous aider à la personnaliser. Deux catégories d'informations peuvent être renvoyées lorsqu'une politique est générée :

- Politique avec informations au niveau de l'action — Pour certains AWS services, tels qu'Amazon EC2, IAM Access Analyzer peut identifier les actions trouvées dans CloudTrail vos événements et répertorie les actions utilisées dans la politique qu'il génère. Pour obtenir la liste des services pris en charge, consultez Services de génération d'une politique d'Analyseur d'accès IAM. Pour certains services, IAM Access Analyzer vous invite à ajouter des actions pour les services à la politique générée.

- Politique avec des informations de niveau service : IAM Access Analyzer utilise les informations relatives aux derniers services consultés pour créer un modèle de politique avec tous les services récemment utilisés. Lorsque vous utilisez le Console de gestion AWS, nous vous invitons à consulter les services et à ajouter des actions pour compléter la politique.

Pour générer une politique basée sur l'activité d'accès

Dans la procédure suivante, nous allons limiter les autorisations accordées à un rôle afin qu'elles correspondent à l'utilisation qu'en fait un utilisateur. Lorsque vous choisissez un utilisateur, choisissez celui dont l'utilisation est représentative du rôle. De nombreux clients créent des comptes d'utilisateurs de test dotés d'PowerUserautorisations, puis leur demandent d'effectuer une série de tâches spécifiques pendant une courte période afin de déterminer quel accès est nécessaire pour effectuer ces tâches,

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d'accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, sélectionnez Utilisateurs, puis choisissez le nom de l'utilisateur pour accéder à la page des détails de l'utilisateur.
4. Dans l'onglet Autorisations, sous Générer une politique basée sur les CloudTrail événements, choisissez Générer une politique.
5. Sur la page Générer une politique, configurez les éléments suivants :
   - Pour Sélectionner une période, sélectionnez Les 7 derniers jours.
   - Pour que le CloudTrail parcours soit analysé, sélectionnez la région et le parcours où l'activité de cet utilisateur est enregistrée.
   - Choisissez Créer et utiliser un nouveau rôle de service.
6. Choisissez Générer une politique, puis attendez que le rôle soit créé. N'actualisez ni ne quittez pas la page de la console tant que le message de notification Politique en cours de génération ne s'affiche pas.
7. Une fois la politique générée, vous devez la revoir et la personnaliser en fonction des besoins du compte IDs et ARNs des ressources. En outre, la politique générée automatiquement peut

ne pas inclure les informations de niveau action nécessaires pour compléter la politique. Pour plus d'informations, consultez la section Génération de politiques IAM Access Analyzer.

Par exemple, modifiez la première instruction qui inclut l'effet `Allow` et l'élément `NotAction` pour n'autoriser que les actions Amazon EC2 et Amazon S3. Pour ce faire, remplacez-la par l'instruction avec l'identifiant `FullAccessToSomeServices`. Votre nouvelle politique peut ressembler à l'exemple de politique suivant.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
          "Sid": "FullAccessToSomeServices",
          "Effect": "Allow",
          "Action": [
              "ec2:*",
              "s3:*"
          ],
          "Resource": "*"
      },
      {
          "Effect": "Allow",
          "Action": [
              "iam:CreateServiceLinkedRole",
              "iam:DeleteServiceLinkedRole",
              "iam:ListRoles",
              "organizations:DescribeOrganization"
          ],
          "Resource": "*"
      }
  ]
}
```

8. Afin d'étayer les bonnes pratiques consistant à octroyer le moindre privilège, examinez et corrigez toutes les erreurs, avertissements ou suggestions renvoyés durant la validation de politique.
9. Pour réduire davantage les autorisations de vos politiques relatives à des actions et à des ressources spécifiques, consultez vos événements dans l'historique des CloudTrail événements. Dans la console, vous pouvez afficher des informations détaillées sur

les actions spécifiques et les ressources auxquelles l'utilisateur a accédé. Pour plus d'informations, consultez la section Affichage CloudTrail des événements dans la CloudTrail console dans le guide de AWS CloudTrail l'utilisateur.

10. Après avoir examiné et validé votre politique, enregistrez-la sous un nom représentatif.
11. Accédez à la page Rôles et choisissez le rôle que les utilisateurs assumeront lorsqu'ils exécuteront les tâches autorisées par votre nouvelle politique.
12. Sélectionnez l'onglet Autorisations, puis choisissez Ajouter des autorisations et sélectionnez Attacher des politiques.
13. Sur la page Attacher les politiques d'autorisation, dans la liste Autres politiques d'autorisation, sélectionnez la politique que vous avez créée, puis choisissez Attacher des politiques.
14. Vous êtes renvoyé à la page des détails du rôle. Deux politiques sont associées au rôle : votre ancienne stratégie AWS gérée, telle que PowerUserAccess, et votre nouvelle stratégie. Cochez la case correspondant à la politique AWS gérée, puis choisissez Supprimer. Lorsque vous êtes invité à confirmer la suppression, choisissez Supprimer.

Les utilisateurs IAM, les principaux fédérés SAML et OIDC et les charges de travail qui assument ce rôle bénéficient désormais d'un accès réduit conformément à la nouvelle politique que vous avez créée.

AWS CLI

Vous pouvez utiliser les commandes suivantes pour générer une politique à l'aide de la AWS CLI.

Pour générer une politique

- analyseur d'accès AWS start-policy-generation

Pour afficher une politique générée

- analyseur d'accès AWS get-generated-policy

Pour annuler une demande de génération de politique

- analyseur d'accès AWS cancel-policy-generation

Pour afficher la liste des demandes de génération de politique

- analyseur d'accès AWS list-policy-generations

API

Vous pouvez utiliser les opérations suivantes pour générer une politique à l'aide de l'API AWS .

Pour générer une politique

- StartPolicyGeneration

Pour afficher une politique générée

- GetGeneratedPolicy

Pour annuler une demande de génération de politique

- CancelPolicyGeneration

Pour afficher la liste des demandes de génération de politique

- ListPolicyGenerations

Utilisation de la recherche pour trouver des ressources IAM

Au fur et à mesure que vous parcourez vos résultats d’accès, vous pouvez utiliser la page de recherche de la console IAM comme option plus rapide pour trouver des ressources IAM. Vous pouvez rechercher des ressources à l’aide de noms de ressources partiels ou d’ARN.

Console

La fonction de recherche de la console IAM vous permet de rechercher tous les éléments suivants :

- Les noms d'entités IAM qui correspondent à vos mots-clés (pour des utilisateurs, des groupes, des rôles, des fournisseurs d'identité et des politiques)
- Les tâches qui correspondent à vos mots-clés

La fonction de recherche de la console IAM ne renvoie pas d'informations sur IAM Access Analyzer.

Chaque ligne des résultats de la recherche est un lien actif. Par exemple, vous pouvez choisir le nom de l'utilisateur dans les résultats de la recherche, ce qui vous dirigera vers la page détaillée de cet utilisateur. Vous pouvez également choisir un lien d'action, par exemple Créer un utilisateur, pour accéder à la page Créer un utilisateur.

> **Note**
>
> La recherche de clé d'accès nécessite que vous saisissiez l'ID de clé d'accès complet dans la zone de recherche. Le résultat de la recherche affiche l'utilisateur associé à cette clé. À partir de là, vous pouvez accéder directement à la page de cet utilisateur, et même y gérer sa clé d'accès.

Utilisez la page Search (Recherche) de la console IAM pour rechercher des éléments associés à ce compte.

Pour rechercher des éléments sur la console IAM

1. Suivez la procédure de connexion adaptée à votre type d'utilisateur, comme décrit dans la rubrique Comment se connecter AWS dans le Guide de l'AWS Sign-In utilisateur.
2. Sur la page d'accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, sélectionnez Search (Recherche).
4. Dans la zone Recherche, saisissez vos mots-clés de recherche.
5. Choisissez un lien dans la liste des résultats de la recherche pour accéder à la partie correspondante de la console.

Les icônes suivantes identifient les types d'éléments obtenus par une recherche :

Icône	Description
	Utilisateurs IAM

Icône	Description
	Groupes IAM
	Rôles IAM
	politiques IAM
	Tâches comme « créer un utilisateur » ou « attacher une politique »
	Résultats obtenus avec le mot-clé **`delete`**

Exemples d'expressions de recherche

Vous pouvez utiliser les expressions suivantes dans la recherche IAM. Remplacez les termes en italique par les noms des utilisateurs, groupes, rôles, clés d'accès, politiques ou fournisseurs d'identité IAM que vous souhaitez rechercher.

- ***user_name*** ou ***group_name*** ou ***role_name*** ou ***policy_name*** ou ***identity_provider_name***
- ***access_key***
- **`add user`** ***user_name*** **`to groups`** ou **`add users to group`** ***group_name***
- **`remove user`** ***user_name*** **`from groups`**
- **`delete`** ***user_name*** ou **`delete`** ***group_name*** ou **`delete`** ***role_name*** ou **`delete`** ***policy_name*** ou **`delete`** ***identity_provider_name***
- **`manage access keys`** ***user_name***
- **`manage signing certificates`** ***user_name***
- **`users`**
- **`manage MFA for`** ***user_name***
- **`manage password for`** ***user_name***
- **`create role`**

- **password policy**
- **edit trust policy for role *role_name***
- **show policy document for role *role_name***
- **attach policy to *role_name***
- **create managed policy**
- **create user**
- **create group**
- **attach policy to *group_name***
- **attach entities to *policy_name***
- **detach entities from *policy_name***

Bonnes pratiques de sécurité et cas d'utilisation dans Gestion des identités et des accès AWS

Gestion des identités et des accès AWS (IAM) propose plusieurs fonctionnalités de sécurité à prendre en compte lors de l'élaboration et de la mise en œuvre de vos propres politiques de sécurité. Les bonnes pratiques suivantes doivent être considérées comme des instructions générales et ne représentent pas une solution de sécurité complète.

Ces pratiques exemplaires peuvent ne pas être appropriées ou suffisantes pour votre environnement spécifique. Par conséquent, considérez-les comme des considérations utiles plutôt que comme des exigences.

Pour tirer le meilleur parti d'IAM, prenez le temps de découvrir les pratiques exemplaires recommandées. Un moyen efficace d'y parvenir est de voir comment IAM est utilisé dans des scénarios réels pour travailler avec d'autres Services AWS.

Rubriques

- Bonnes pratiques de sécurité dans IAM
- Meilleures pratiques en matière d'utilisateur root pour votre Compte AWS
- Cas d'utilisation métier pour IAM

Bonnes pratiques de sécurité dans IAM

Follow us on Twitter

Pour vous aider à sécuriser vos AWS ressources, suivez ces bonnes pratiques pour Gestion des identités et des accès AWS (IAM).

Après avoir mis en œuvre ces bonnes pratiques, consultez d'autres sujets Sécurité dans IAM et AWS STS de sécurité, notamment la validation de la conformité, la résilience et la sécurité de l'infrastructure.

Rubriques

- Obliger les utilisateurs humains à utiliser la fédération avec un fournisseur d'identité pour accéder AWS en utilisant des informations d'identification temporaires
- Exiger que les charges de travail utilisent des informations d'identification temporaires avec des rôles IAM pour accéder AWS

- Authentification multifactorielle (MFA) nécessaire.
- Mettre à jour les clés d'accès lorsque cela est nécessaire pour les cas d'utilisation nécessitant des informations d'identification à long terme
- Suivez les meilleures pratiques pour protéger vos informations d'identification de l'utilisateur root
- Accorder les autorisations de moindre privilège
- Commencez avec AWS politiques gérées et évolution vers les autorisations de moindre privilège
- Utiliser IAM Access Analyzer pour générer des politiques de moindre privilège en fonction de l'activité d'accès
- Vérifiez et supprimez régulièrement les utilisateurs, les rôles, les autorisations, les politiques et les informations d'identification non utilisés
- Utiliser des conditions dans les stratégies IAM pour restreindre davantage l'accès
- Vérifiez l'accès public et intercompte aux ressources avec IAM Access Analyzer.
- Utilisez IAM Access Analyzer pour valider vos politiques IAM pour des autorisations sécurisées et fonctionnelles
- Établir des garde-fous d'autorisations sur plusieurs comptes
- Utiliser les limites des autorisations pour déléguer la gestion des autorisations au sein d'un compte.

Obliger les utilisateurs humains à utiliser la fédération avec un fournisseur d'identité pour accéder AWS en utilisant des informations d'identification temporaires

Les utilisateurs humains, également connus sous le nom identités humaines, sont les personnes, les administrateurs, les développeurs, les opérateurs et les consommateurs de vos applications. Ils doivent disposer d'une identité pour accéder à vos AWS environnements et applications. Les utilisateurs humains membres de votre organisation sont également appelés Identités de personnel. Les utilisateurs humains peuvent également être des utilisateurs externes avec lesquels vous collaborez et qui interagissent avec vos AWS ressources. Ils peuvent le faire via un navigateur Web, une application client, une application mobile ou des outils de ligne de commande interactifs.

Demandez à vos utilisateurs humains d'utiliser des informations d'identification temporaires lors de l'accès AWS. Vous pouvez utiliser un fournisseur d'identité pour que vos utilisateurs humains fournissent un accès fédéré Comptes AWS en assumant des rôles, qui fournissent des informations d'identification temporaires. Pour une gestion centralisée des accès, nous vous recommandons d'utiliser AWS IAM Identity Center (IAM Identity Center) pour gérer l'accès à vos comptes et les

autorisations au sein de ceux-ci. Vous pouvez gérer vos identités d'utilisateur avec IAM Identity Center ou gérer les autorisations d'accès pour les identités des utilisateurs dans IAM Identity Center à partir d'un fournisseur d'identité externe. Pour plus d'informations, consultez Présentation de AWS IAM Identity Center dans le Guide de l'utilisateur AWS IAM Identity Center .

Vous pouvez utiliser le gestionnaire d'accès aux comptes, une fonctionnalité IAM qui vous permet d'attribuer des rôles IAM existants sur les comptes de votre organisation aux utilisateurs et aux groupes IAM Identity Center. Le gestionnaire d'accès aux comptes vous donne accès à l'ensemble des fonctionnalités des rôles IAM, y compris des politiques de confiance personnalisées, des balises de rôle pour ABAC et des chemins de rôle configurables. Vous pouvez l'utiliser conjointement avec des ensembles d'autorisations ou seul.

Pour plus d'informations sur les rôles , consultez Termes et concepts relatifs aux rôles.

Exiger que les charges de travail utilisent des informations d'identification temporaires avec des rôles IAM pour accéder AWS

Une charge de travail est un ensemble de ressources et de code qui fournit une valeur business, par exemple une application destinée au client ou un processus de backend. Votre charge de travail peut comporter des applications, des outils opérationnels et des composants qui nécessitent des informations d'identification pour envoyer des demandes à Services AWS, telles que les demandes de lecture de données depuis Amazon S3.

Lorsque vous utilisez un service de AWS calcul, tel qu'Amazon EC2 ou Lambda, AWS fournit les informations d'identification temporaires d'un rôle IAM à cette ressource de calcul. Les applications écrites à l'aide d'un AWS SDK découvriront et utiliseront ces informations d'identification temporaires pour accéder aux AWS ressources, et il n'est pas nécessaire de distribuer des informations d'identification de longue durée pour un utilisateur IAM à vos charges de travail exécutées. AWS

Les charges de travail qui s'exécutent en dehors de AWS, telles que vos serveurs sur site, les serveurs d'autres fournisseurs de cloud ou les plateformes gérées d'intégration continue et de livraison continue (CI/CD), peuvent toujours utiliser des informations d'identification temporaires. Cependant, vous devrez fournir ces informations d'identification temporaires à votre charge de travail. Voici plusieurs façons de fournir des informations d'identification temporaires à vos charges de travail :

- Vous pouvez utiliser IAM Roles Anywhere pour demander des AWS informations d'identification temporaires pour votre charge de travail à l'aide d'un X.509 certificat provenant de votre infrastructure à clé publique (PKI).

- Vous pouvez appeler l' AWS STS `AssumeRoleWithSAML`API pour demander des AWS informations d'identification temporaires pour votre charge de travail à l'aide d'une assertion SAML provenant d'un fournisseur d'identité externe (IdP) configuré dans votre. Compte AWS
- Vous pouvez appeler l' AWS STS `AssumeRoleWithWebIdentity`API pour demander des AWS informations d'identification temporaires pour votre charge de travail à l'aide d'un jeton Web JSON (JWT) provenant d'un IdP configuré dans votre. Compte AWS
- Vous pouvez demander des AWS informations d'identification temporaires à votre appareil IoT à l'aide de l'authentification MTLS (Mutual Transport Layer Security) à l'aide AWS IoT Core de.

Certains prennent Services AWS également en charge des intégrations pour fournir des informations d'identification temporaires à vos charges de travail exécutées en dehors de : AWS

- Amazon Elastic Container Service (Amazon Anywhere ECS) vous permet d'exécuter des tâches Amazon ECS sur vos propres ressources de calcul et fournit des informations d'identification AWS temporaires à vos tâches Amazon ECS s'exécutant sur ces ressources de calcul.
- Amazon Elastic Kubernetes Service vous Hybrid Nodes permet de joindre vos ressources de calcul exécutées en dehors des AWS nœuds à un cluster Amazon EKS. Amazon EKS peut fournir des informations d'identification temporaires aux pods Amazon EKS s'exécutant sur vos ressources de calcul.
- AWS Systems ManagerHybrid Activationsvous permet de gérer vos ressources de calcul exécutées en dehors de AWS SSM et fournit des AWS informations d'identification temporaires à l'agent SSM qui s'exécute sur vos ressources de calcul.

Authentification multifactorielle (MFA) nécessaire.

Nous recommandons d'utiliser les rôles IAM pour les utilisateurs humains et les charges de travail qui accèdent à vos ressources AWS afin qu'ils utilisent des informations d'identification temporaires. Toutefois, pour les scénarios dans lesquels vous avez besoin d'un utilisateur IAM ou root dans votre compte, exigez une MFA pour plus de sécurité. Avec MFA, les utilisateurs disposent d'un périphérique qui génère une réponse à une stimulation d'authentification. Les informations d'identification de l'utilisateur et la réponse générée par le périphérique sont requis pour mener à bien le processus du connexion. Pour de plus amples informations, veuillez consulter AWS Multi-factor authentification dans IAM. Nous vous recommandons d'utiliser une authentification multifacteur résistante au phishing, telle que des clés d'accès et des clés de sécurité, dans la mesure du possible.

Pour de plus amples informations, veuillez consulter Attribuez un mot de passe ou une clé de sécurité dans le Console de gestion AWS.

Si vous utilisez IAM Identity Center pour une gestion centralisée des accès des utilisateurs humains, vous pouvez utiliser les capacités MFA IAM Identity Center lorsque votre source d'identités est configurés avec le magasin d'identités, Microsoft AD géré par AWS ou AD Connector. Pour plus d'informations sur l'authentification multifacteur dans IAM Identity Center, consultez le Guide de l'AWS IAM Identity Center utilisateur consacré à l'Multi-factorauthentification.

Mettre à jour les clés d'accès lorsque cela est nécessaire pour les cas d'utilisation nécessitant des informations d'identification à long terme

Dans la mesure du possible, nous vous recommandons de vous appuyer sur des informations d'identification temporaires plutôt que de créer des informations d'identification à long terme tels que les clés d'accès. Toutefois, pour les scénarios dans lesquels vous avez besoin d'utilisateurs IAM disposant d'un accès par programmation et d'informations d'identification à long terme, nous vous recommandons de mettre à jour les clés d'accès lorsque cela est nécessaire, par exemple lorsqu'un employé quitte votre entreprise. Nous vous recommandons d'utiliser les dernières informations consultées relatives à un accès IAM pour mettre à jour et retirer les clés d'accès en toute sécurité. Pour de plus amples informations, veuillez consulter Mise à jour des clés d'accès.

Certains cas d'utilisation spécifiques nécessitent des informations d'identification à long terme auprès des utilisateurs IAM. AWS Voici certaines des fonctionnalités les plus utilisées :

- Charges de travail qui ne peuvent pas utiliser de rôles IAM — Vous pouvez exécuter une charge de travail à partir d'un emplacement qui a besoin d'accéder à AWS. Dans certains cas, vous ne pouvez pas utiliser les rôles IAM pour fournir des informations d'identification temporaires, par exemple pour les WordPress plugins. Dans ces situations, utilisez les clés d'accès à long terme de l'utilisateur IAM pour cette charge de travail afin de vous authentifier auprès de AWS.
- Third-party AWS clients : si vous utilisez des outils qui ne prennent pas en charge l'accès à IAM Identity Center, tels que des AWS clients tiers ou des fournisseurs qui ne sont pas hébergés sur AWS, utilisez des clés d'accès utilisateur à long terme IAM.
- AWS CodeCommit accès : si vous l'utilisez CodeCommit pour stocker votre code, vous pouvez utiliser un utilisateur IAM avec des clés SSH ou des informations d'identification spécifiques au service pour vous authentifier CodeCommit auprès de vos référentiels. Nous vous recommandons de procéder ainsi en plus de vous servir d'un utilisateur dans IAM Identity Center pour une authentification ordinaire. Les utilisateurs d'IAM Identity Center sont les membres de votre

personnel qui ont besoin d'accéder à vos applications Comptes AWS ou à vos applications cloud. Pour permettre aux utilisateurs d'accéder à vos CodeCommit référentiels sans configurer les utilisateurs IAM, vous pouvez configurer l'git-remote-codecommitutilitaire. Pour plus d'informations sur IAM et CodeCommit, consultezInformations d'identification IAM pour CodeCommit : informations d'identification Git, clés SSH et AWS clés d'accès. Pour plus d'informations sur la configuration de l'git-remote-codecommitutilitaire, consultez la section Connexion à AWS CodeCommit des référentiels avec des informations d'identification rotatives dans le Guide de AWS CodeCommit l'utilisateur.

- Amazon Keyspaces (for Apache Cassandra) access (Accès Amazon Keyspaces (pour Apache Cassandra)) : dans une situation où vous n'êtes pas en mesure de vous servir des utilisateurs dans IAM Identity Center, par exemple à des fins de test de compatibilité avec Cassandra, vous pouvez utiliser un utilisateur IAM avec des informations d'identification spécifiques au service pour vous authentifier auprès d'Amazon Keyspaces. Les utilisateurs d'IAM Identity Center sont les membres de votre personnel qui ont besoin d'accéder à vos applications Comptes AWS ou à vos applications cloud. Vous pouvez également vous connecter à Amazon Keyspaces à l'aide d'informations d'identification temporaires. Pour de plus d'informations, veuillez consulter Utilisation d'informations d'identification temporaires pour se connecter à Amazon Keyspaces à l'aide d'un rôle IAM et du plugin Sigv4 dans le Guide du développeur Amazon Keyspaces (pour Apache Cassandra).

Suivez les meilleures pratiques pour protéger vos informations d'identification de l'utilisateur root

Lorsque vous créez un Compte AWS, vous définissez les informations d'identification de l'utilisateur root pour vous connecter au Console de gestion AWS. Protégez vos informations d'identification d'utilisateur root de la même manière que vous protégez d'autres informations personnelles sensibles. Pour mieux comprendre comment sécuriser et mettre à l'échelle vos processus d'utilisateur root, consultez Meilleures pratiques en matière d'utilisateur root pour votre Compte AWS.

Accorder les autorisations de moindre privilège

Lorsque vous définissez des autorisations avec des stratégies IAM, accordez uniquement les autorisations nécessaires à l'exécution d'une seule tâche. Pour ce faire, vous définissez les actions qui peuvent être entreprises sur des ressources spécifiques dans des conditions spécifiques, également appelées autorisations de moindre privilège. Vous pouvez commencer par des autorisations étendues tout en explorant les autorisations requises pour votre charge de travail ou

votre cas d'utilisation. Au fur et à mesure que votre cas d'utilisation évolue, vous pouvez réduire les autorisations que vous accordez pour obtenir le moindre privilège. Pour de plus d'informations sur l'utilisation de IAM pour appliquer des autorisations, consultez Politiques et autorisations dans Gestion des identités et des accès AWS.

Commencez avec AWS politiques gérées et évolution vers les autorisations de moindre privilège

Pour commencer à accorder des autorisations à vos utilisateurs et charges de travail, utilisez les AWS politiques gérées qui accordent des autorisations dans de nombreux cas d'utilisation courants. Ils sont disponibles dans votre Compte AWS. N'oubliez pas que les politiques AWS gérées peuvent ne pas accorder les autorisations de moindre privilège pour vos cas d'utilisation spécifiques, car elles sont disponibles pour tous AWS les clients. En conséquence, nous vous recommandons de réduire encore les autorisations en définissant des Politiques gérées par le client qui sont spécifiques à vos cas d'utilisation. Pour de plus amples informations, veuillez consulter AWS stratégies gérées. Pour plus d'informations sur les politiques AWS gérées conçues pour des fonctions spécifiques, consultezAWS politiques gérées pour les fonctions professionnelles.

Utiliser IAM Access Analyzer pour générer des politiques de moindre privilège en fonction de l'activité d'accès

Pour accorder uniquement les autorisations nécessaires à l'exécution d'une seule tâche, vous pouvez générer des politiques en fonction de votre activité d'accès connectée AWS CloudTrail. IAM Access Analyzer analyse les services et les actions que vos rôles IAM utilisent, puis génère une stratégie précise que vous pouvez utiliser. Après avoir testé chaque stratégie générée, vous pouvez la déployer dans votre environnement de production. Cela garantit que vous n'accordez que les autorisations requises à vos charges de travail. Pour plus d'informations sur la génération de politiques, consultez IAM Access Analyzer policy generation.

Vérifiez et supprimez régulièrement les utilisateurs, les rôles, les autorisations, les politiques et les informations d'identification non utilisés

Vous avez peut-être des utilisateurs, des rôles, des autorisations, des stratégies ou des informations d'identification IAM dont vous n'avez plus besoin dans votre Compte AWS. IAM fournit dernière information consultées pour vous aider à identifier les utilisateurs, les rôles, les autorisations, les stratégies et les informations d'identification dont vous n'avez plus besoin afin de pouvoir les supprimer. Cela vous permet de réduire le nombre d'utilisateurs, de rôles, d'autorisations, de

politiques et d'informations d'identification que vous devez surveiller. Vous pouvez ainsi peaufiner vos politiques IAM afin qu'elles respectent au plus près le principe du moindre privilège. Pour de plus amples informations, veuillez consulter Affiner les autorisations en AWS utilisant les dernières informations consultées.

Utiliser des conditions dans les stratégies IAM pour restreindre davantage l'accès

Vous pouvez spécifier les conditions dans lesquelles une déclaration de politique est en vigueur. Ainsi, vous pouvez accorder l'accès aux actions et aux ressources, mais uniquement si la demande d'accès répond à des conditions spécifiques. Par exemple, vous pouvez écrire une condition de politique pour spécifier que toutes les demandes doivent être envoyées via TLS. Vous pouvez également utiliser des conditions pour accorder l'accès aux actions de service, mais uniquement si elles sont utilisées par le biais d'un outil spécifique Service AWS, tel que CloudFormation. Pour de plus amples informations, veuillez consulter Éléments de la politique JSON IAM : Condition.

Vérifiez l'accès public et intercompte aux ressources avec IAM Access Analyzer.

Avant d'accorder des autorisations d'accès public ou intercomptes à AWS, nous vous recommandons de vérifier si un tel accès est requis. Vous pouvez utiliser IAM Access Analyzer pour vous aider à prévisualiser et à analyser l'accès public et intercompte pour les types de ressources pris en charge. Pour ce faire, consultez les résultats que génère IAM Access Analyzer. Ces résultats vous aident à vérifier que vos contrôles d'accès aux ressources accordent l'accès que vous attendez. En outre, lorsque vous mettez à jour les autorisations publiques et entre comptes, vous pouvez vérifier l'effet de vos modifications avant de déployer de nouveaux contrôles d'accès à vos ressources. IAM Access Analyzer surveille également en permanence les types de ressources pris en charge et génère une recherche pour les ressources qui autorisent un accès public ou intercompte. Pour plus d'informations, consultez l'onglet Prévisualisation de l'accès avec les API IAM Access Analyzer.

Utilisez IAM Access Analyzer pour valider vos politiques IAM pour des autorisations sécurisées et fonctionnelles

Validez les politiques que vous créez pour vous assurer qu'elles respectent le langage de politique IAM (JSON) et les meilleures pratiques IAM. Vous pouvez valider vos politiques à l'aide de la vérification de politique IAM Access Analyzer. IAM Access Analyzer fournit plus de 100 vérifications de politiques et des recommandations exploitables pour vous aider à créer des politiques sécurisées

et fonctionnelles. Lorsque vous créez de nouvelles politiques ou que vous modifiez des politiques existantes dans la console, IAM Access Analyzer fournit des recommandations pour vous aider à affiner et à valider vos politiques avant de les enregistrer. En outre, nous vous recommandons d'examiner et de valider toutes vos politiques existantes. Pour de plus d'informations, veuillez consulter l'onglet Validation de politique IAM Access Analyzer. Pour en savoir plus sur les vérifications de politique fournies par IAM Access Analyzer, veuillez consulter IAM Access Analyzer policy validation (Validation de politique IAM Access Analyzer).

Établir des garde-fous d'autorisations sur plusieurs comptes

Au fur et à mesure que vous adaptez vos charges de travail, séparez-les en utilisant plusieurs comptes gérés avec AWS Organizations. Nous vous recommandons d'utiliser des politiques de contrôle des AWS Organizations services (SCP) pour établir des barrières d'autorisations afin de contrôler l'accès de tous les principaux acteurs (rôles IAM et utilisateurs) à vos comptes. Nous vous recommandons d'utiliser les politiques de contrôle des ressources (RCP) AWS Organizations pour établir des garde-fous d'autorisations afin de contrôler l'accès aux ressources AWS de votre organisation. Les SCP et les RCP sont des types de politiques d'organisation que vous pouvez utiliser pour gérer les autorisations dans votre organisation au niveau de l' AWS organisation, de l'unité organisationnelle (OU) ou du compte.

Toutefois, les SCP et les RCP ne suffisent pas à accorder des autorisations aux principaux et aux ressources de votre organisation. Aucune autorisation n'est accordée par les SCP et les RCP. Pour octroyer des autorisations, votre administrateur doit toujours attacher des politiques basées sur les ressources ou basées sur l'identité aux utilisateurs IAM, aux rôles IAM, ou aux ressources de vos comptes. Pour plus d'informations, consultez la section Éléments constitutifs — AWS Organizations, comptes et garde-corps SRA.

Utiliser les limites des autorisations pour déléguer la gestion des autorisations au sein d'un compte.

Dans certains scénarios, vous souhaitez peut-être déléguer la gestion des autorisations d'un compte à d'autres. Par exemple, vous pouvez autoriser les développeurs à créer et à gérer des rôles pour leurs charges de travail. Lorsque vous déléguez des autorisations à d'autres personnes, utilisez Limites d'autorisations pour définir les autorisations maximales que vous déléguez. Une limite d'autorisations est une fonctionnalité avancée permettant d'utiliser une politique gérée pour définir les autorisations maximales qu'une politique basée sur l'identité peut accorder à une rôle IAM. Une limite d'autorisations restreint le nombre maximum d'autorisations, mais n'accorde pas seule l'accès. Pour de plus amples informations, veuillez consulter Limites d'autorisations pour les entités IAM.

Meilleures pratiques en matière d'utilisateur root pour votre Compte AWS

Lorsque vous créez un Compte AWS, vous commencez par un ensemble d'informations d'identification par défaut avec un accès complet à toutes les AWS ressources de votre compte. Cette identité est appelée l'utilisateur root Compte AWS. Nous vous recommandons vivement de ne pas accéder à l'utilisateur Compte AWS root à moins que votre tâche ne nécessite des informations d'identification d'utilisateur root. Vous devez sécuriser vos informations d'identification d'utilisateur root et les mécanismes de récupération de votre compte pour vous assurer de ne pas exposer vos informations d'identification hautement privilégiées à des fins d'utilisation non autorisée.

Pour les comptes Comptes AWS gérés via plusieurs comptes AWS Organizations, nous vous recommandons de supprimer les informations d'identification de l'utilisateur root des comptes membres afin d'empêcher toute utilisation non autorisée. Vous pouvez supprimer le mot de passe d'utilisateur racine, les clés d'accès, les certificats de signature, ainsi que désactiver et supprimer l'authentification multifactorielle (MFA). Les comptes membres ne peuvent pas se connecter à leur utilisateur racine ni récupérer le mot de passe de leur utilisateur racine. Pour de plus amples informations, veuillez consulter Gestion centralisée de l'accès root pour les comptes membres.

Au lieu d'accéder à l'utilisateur root, créez un utilisateur administratif pour les tâches quotidiennes.

- Si vous en avez un nouveau Compte AWS, voyezConfiguration de votre Compte AWS.
- Pour la Comptes AWS gestion multiple via AWS Organizations, voir Configuration de Compte AWS l'accès pour un utilisateur administratif d'IAM Identity Center.

Avec votre utilisateur administratif, vous pouvez ensuite créer des identités supplémentaires pour les utilisateurs qui ont besoin d'accéder aux ressources de votre Compte AWS. Nous vous recommandons vivement de demander aux utilisateurs de s'authentifier à l'aide d'informations d'identification temporaires lors de l'accès AWS.

- Pour un compte unique et autonome Compte AWS, utilisez-le Rôles IAM pour créer des identités dans votre compte avec des autorisations spécifiques. Les rôles sont destinés à être endossés par toute personne qui en a besoin. En outre, un rôle ne dispose pas d'informations d'identification standard à long terme comme un mot de passe ou des clés d'accès associées. Au lieu de cela, lorsque vous adoptez un rôle, il vous fournit des informations d'identification de sécurité temporaires pour votre session de rôle. Contrairement aux rôles IAM, Utilisateurs IAM dispose d'informations d'identification à long terme telles que des mots de passe et des clés d'accès. Dans

la mesure du possible, nous vous recommandons, dans le cadre des bonnes pratiques, de vous appuyer sur des informations d'identification temporaires plutôt que de créer des utilisateurs IAM ayant des informations d'identification à long terme tels que les clés d'accès.

- Pour plusieurs utilisateurs Comptes AWS gérés via AWS Organizations, utilisez IAM Identity Center Workforce Users. Avec IAM Identity Center, vous pouvez gérer de manière centralisée les utilisateurs de vos comptes Comptes AWS et les autorisations associées à ces comptes. Gérez vos identités d'utilisateur à l'aide d'IAM Identity Center ou depuis un fournisseur d'identité externe. Pour plus d’informations, consultez Présentation de AWS IAM Identity Center dans le Guide de l’utilisateur AWS IAM Identity Center .

Rubriques

- Sécurisez vos informations d’identification d'utilisateur root pour empêcher toute utilisation non autorisée
- Utiliser un mot de passe utilisateur root fort pour protéger l'accès
- Sécurisez votre connexion d'utilisateur root avec l'authentification multifactorielle (MFA)
- Ne créer aucune clé d'accès pour l'utilisateur root
- Utiliser une approbation par plusieurs personnes pour la connexion de l'utilisateur root dans la mesure du possible
- Utiliser une adresse e-mail de groupe pour les informations d'identification de l'utilisateur root
- Restreindre l'accès aux mécanismes de récupération des comptes
- Sécurisez votre AWS Organizations informations d'identification de l'utilisateur root du compte
- Surveiller l'accès et l'utilisation

Sécurisez vos informations d’identification d'utilisateur root pour empêcher toute utilisation non autorisée

Sécurisez vos informations d’identification d'utilisateur root et ne les utilisez que pour les tâches qui les nécessitent. Pour empêcher toute utilisation non autorisée, ne communiquez pas votre mot de passe d'utilisateur root, votre MFA, vos clés d'accès, vos paires de CloudFront clés ou vos certificats de signature à quiconque, sauf à ceux qui ont un besoin professionnel strict d'accéder à l'utilisateur root.

Ne stockez pas le mot de passe de l'utilisateur root avec des outils qui dépendent d'un compte accessible Services AWS à l'aide de ce même mot de passe. En cas de perte ou d’oubli de votre mot

de passe d’utilisateur racine, vous ne pourrez pas accéder à ces outils. Nous vous recommandons de donner la priorité à la résilience et d'envisager de demander à deux personnes ou plus d'autoriser l'accès à l'emplacement de stockage. L'accès au mot de passe ou à son emplacement de stockage doit être consigné et surveillé.

Utiliser un mot de passe utilisateur root fort pour protéger l'accès

Nous vous recommandons d'utiliser un mot de passe fort et unique. Des outils tels que des gestionnaires de mots de passe dotés d'algorithmes de génération de mots de passe puissants peuvent vous permettre d'atteindre ces objectifs. AWS exige que votre mot de passe remplisse les conditions suivantes :

- Avoir un minimum de 8 caractères et un maximum de 128 caractères
- Inclure au minimum trois des types de caractères suivants : majuscules, minuscules, chiffres, et les symboles ! @ # $ % ^ & * () <> [] {} | _ + - =
- Il ne doit pas être identique à votre Compte AWS nom ou à votre adresse e-mail.

Pour de plus amples informations, veuillez consulter Changez le mot de passe pour Utilisateur racine d'un compte AWS.

Sécurisez votre connexion d'utilisateur root avec l'authentification multifactorielle (MFA)

Étant donné qu'un utilisateur root peut effectuer des actions privilégiées, il est essentiel d'ajouter MFA pour l'utilisateur root comme deuxième facteur d'authentification, en plus de l'adresse e-mail et du mot de passe comme informations d'identification de connexion. Vous pouvez enregistrer jusqu'à huit appareils MFA de n'importe quelle combinaison des types MFA actuellement pris en charge auprès de votre utilisateur Compte AWS root.

Nous vous recommandons fortement d’activer plusieurs dispositifs MFA pour vos informations d’identification d’utilisateur racine afin de renforcer la flexibilité et la résilience de votre politique de sécurité. Tous les types d’ Compte AWS (comptes autonomes, comptes de gestion et comptes membres) nécessitent la configuration de l’authentification multifacteur (MFA) pour leur utilisateur racine. Les utilisateurs doivent enregistrer l’authentification multifactorielle (MFA) dans les 35 jours suivant leur première tentative de connexion à l’ Console de gestion AWS si la MFA n’est pas déjà activée.

- Les clés de sécurité matérielles certifiées FIDO sont fournies par des fournisseurs tiers. Pour plus d'informations, voir Activer une clé de sécurité FIDO pour l'utilisateur Compte AWS root.
- Un périphérique matériel qui génère un code numérique à six chiffres basé sur l'algorithme TOTP (mot de passe unique à durée limitée). Pour plus d'informations, voir Activer un jeton TOTP matériel pour l'utilisateur Compte AWS root.
- Une appli d'authentification virtuelle qui s'exécute sur un téléphone ou un autre appareil et qui égale le niveau d'un dispositif physique. Pour plus d'informations, consultez la section Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root.

Ne créer aucune clé d'accès pour l'utilisateur root

Les touches d'accès vous permettent d'exécuter des commandes dans l'interface de ligne de AWS commande (AWS CLI) ou d'utiliser des opérations d'API à partir de l'un des AWS kits SDK. Nous vous recommandons vivement de ne pas créer de clés d'accès pour votre utilisateur root, car celui-ci dispose d'un accès complet à toutes Services AWS les ressources du compte, y compris aux informations de facturation.

Étant donné que seules quelques tâches nécessitent l'utilisation de l'utilisateur root et que vous ne les effectuez généralement que rarement, nous vous recommandons de vous connecter au pour Console de gestion AWS effectuer les tâches de l'utilisateur root. Pour l'accès par programmation, au lieu de créer des clés d'accès pour l'utilisateur root, utilisez la `aws login` commande avec vos informations d'identification root pour vous authentifier auprès des SDK AWS CLI et. Cette approche fournit des informations d'identification temporaires et automatiquement remplacées qui améliorent votre niveau de sécurité tout en éliminant la nécessité de gérer des clés d'accès à long terme. Avant de créer des clés d'accès, passez en revueAlternatives aux clés d'accès à long terme.

Utiliser une approbation par plusieurs personnes pour la connexion de l'utilisateur root dans la mesure du possible

Envisagez d'utiliser l'approbation par plusieurs personnes pour vous assurer que personne ne peut accéder à la fois au MFA et au mot de passe de l'utilisateur root. Certaines entreprises ajoutent une couche de sécurité supplémentaire en configurant un groupe d'administrateurs ayant accès au mot de passe et un autre groupe d'administrateurs ayant accès à la MFA. Un membre de chaque groupe doit se réunir pour se connecter en tant qu'utilisateur root.

Utiliser une adresse e-mail de groupe pour les informations d'identification de l'utilisateur root

Utilisez une adresse e-mail gérée par votre entreprise et transférez les messages reçus directement à un groupe d'utilisateurs. Si vous AWS devez contacter le titulaire du compte, cette approche réduit le risque de retard dans la réponse, même si les personnes sont en vacances, sont en congé de maladie ou ont quitté l'entreprise. L'adresse e-mail utilisée pour l'utilisateur root ne doit pas être utilisée à d'autres fins.

Restreindre l'accès aux mécanismes de récupération des comptes

Assurez-vous de développer un processus pour gérer les mécanismes de récupération des informations d'identification des utilisateurs root au cas où vous auriez besoin d'y accéder en cas d'urgence, par exemple en cas de prise de contrôle de votre compte administratif.

- Assurez-vous d'avoir accès à la boîte de réception de votre utilisateur root afin de pouvoir réinitialiser le mot de passe d'un utilisateur root perdu ou oublié.
- Si le MFA de votre utilisateur Compte AWS root est perdu, endommagé ou ne fonctionne pas, vous pouvez vous connecter à l'aide d'un autre MFA enregistré avec les mêmes informations d'identification d'utilisateur root. Si vous avez perdu l’accès à toutes vos MFA, vous avez besoin que le numéro de téléphone et l’e-mail utilisés pour enregistrer votre compte soient à jour et accessibles pour récupérer votre MFA. Pour de plus amples informations, consultez la section Récupération d'un dispositif MFA d'utilisateur root.
- Si vous choisissez de ne pas stocker votre mot de passe d'utilisateur root et votre MFA, le numéro de téléphone enregistré dans votre compte peut être utilisé comme autre moyen de récupérer les informations d'identification d'utilisateur root. Assurez-vous d'avoir accès au numéro de téléphone du contact, mettez-le à jour et limitez le nombre de personnes autorisées à gérer le numéro de téléphone.

Personne ne doit avoir accès à la fois à la boîte de réception de messagerie et au numéro de téléphone, car les deux sont des canaux de vérification permettant de récupérer votre mot de passe d'utilisateur root. Il est important que deux groupes de personnes gèrent ces canaux. Un groupe ayant accès à votre adresse e-mail principale et un autre groupe ayant accès au numéro de téléphone principal pour récupérer l'accès à votre compte en tant qu'utilisateur root.

Sécurisez votre AWS Organizations informations d'identification de l'utilisateur root du compte

Lorsque vous passez à une stratégie multi-comptes avec AWS Organizations, chacun de vos comptes Comptes AWS possède ses propres informations d'identification utilisateur root que vous devez sécuriser. Le compte que vous utilisez pour créer votre organisation est le compte de gestion et les autres comptes de votre organisation sont des comptes membres.

Sécuriser les informations d’identification de l’utilisateur racine pour le compte de gestion

AWS vous oblige à enregistrer le MFA pour l'utilisateur root du compte de gestion de votre organisation. L’enregistrement MFA doit être effectué lors de la première tentative de connexion ou dans le délai de grâce de 35 jours. Si la MFA n’est pas activée dans ce délai, vous devrez vous inscrire avant de pouvoir accéder à l’ Console de gestion AWS. Pour de plus amples informations, veuillez consulter Multi-factor authentification pour Utilisateur racine d'un compte AWS.

Sécuriser les informations d'identification d'utilisateur root pour les comptes membres

Si vous gérez AWS Organizations plusieurs comptes, vous pouvez adopter deux stratégies pour sécuriser l'accès des utilisateurs root à votre compte AWS Organizations.

- Centralisez l’accès racine et supprimez les informations d’identification des utilisateurs racines des comptes membres. Supprimez les informations d’identification de l’utilisateur racine, les clés d’accès, les certificats de signature, ainsi que désactivez et supprimez l’authentification multifactorielle (MFA). Lorsque cette politique est utilisée, les comptes membres ne peuvent pas se connecter à leur utilisateur racine ni récupérer le mot de passe de leur utilisateur racine. Pour de plus amples informations, veuillez consulter Gestion centralisée de l’accès root pour les comptes membres.
- Sécurisez les informations d'identification des utilisateurs root de vos AWS Organizations comptes grâce à l'authentification multifacteur pour renforcer la sécurité des comptes. Pour de plus amples informations, veuillez consulter Multi-factor authentification pour Utilisateur racine d'un compte AWS.

Pour plus de détails, consultez la section Accès aux comptes membres de votre organisation dans le AWS Organizations Guide de l’utilisateur .

Définissez des contrôles de sécurité préventifs dans AWS Organizations à l'aide d'une politique de contrôle des services (SCP)

Si les comptes membres de votre organisation disposent d'informations d'identification de l'utilisateur racine, vous pouvez appliquer un SCP pour restreindre l'accès à l'utilisateur racine du compte membre. Le fait de refuser toutes les actions de l'utilisateur root sur vos comptes membres, à l'exception de certaines actions réservées au root, permet d'empêcher tout accès non autorisé. Pour plus de détails, consultez la section Utiliser une politique de contrôle des services (SCP) pour restreindre ce que l'utilisateur root de vos comptes membres peut faire.

Surveiller l'accès et l'utilisation

Nous vous recommandons d'utiliser vos mécanismes de suivi actuels pour surveiller, alerter et signaler la connexion et l'utilisation des informations d'identification d'utilisateur root, y compris les alertes annonçant la connexion et l'utilisation de l'utilisateur root. Les services suivants peuvent vous aider à suivre l'utilisation des informations d'identification de l'utilisateur root et à effectuer des contrôles de sécurité afin d'empêcher toute utilisation non autorisée.

> **Note**
>
> CloudTrail enregistre différents événements de connexion pour les sessions de l'utilisateur root et des utilisateurs root privilégiés. Ces sessions privilégiées permettent d'effectuer des tâches qui nécessitent des informations d'identification de l'utilisateur racine dans les comptes membres de votre organisation. Vous pouvez utiliser l'événement de connexion pour identifier les actions effectuées par le compte de gestion ou un administrateur délégué à l'aide de `sts:AssumeRoot`. Pour de plus amples informations, veuillez consulter Suivi des tâches privilégiées dans CloudTrail.

- Si vous souhaitez être informé de l'activité de connexion de l'utilisateur root sur votre compte, vous pouvez utiliser Amazon CloudWatch pour créer une règle d'événements qui détecte l'utilisation des informations d'identification de l'utilisateur root et déclenche une notification à votre administrateur de sécurité. Pour plus de détails, voir Surveiller et notifier l'activité de l'utilisateur Compte AWS root.
- Si vous souhaitez configurer des notifications pour vous avertir des actions approuvées de l'utilisateur root, vous pouvez utiliser Amazon et Amazon EventBridge SNS pour rédiger une EventBridge règle permettant de suivre l'utilisation de l'utilisateur root pour l'action spécifique et

vous avertir en utilisant une rubrique Amazon SNS. Pour obtenir un exemple, consultez Envoyer une notification lorsqu'un objet Amazon S3 est créé.

- Si vous l'utilisez déjà GuardDuty comme service de détection des menaces, vous pouvez étendre sa capacité pour vous avertir lorsque les informations d'identification de l'utilisateur root sont utilisées sur votre compte.

Cette alerte doit inclure, sans s'y limiter, l'adresse e-mail pour l'utilisateur root. Vérifiez que vous avez des procédures en place pour savoir comment répondre aux alertes afin que le personnel qui reçoit une alerte d'accès d'utilisateur root comprenne comment confirmer que l'accès de l'utilisateur root est attendu et comment remonter l'événement s'il croit qu'un incident de sécurité est en cours. Pour un exemple de configuration des alertes, voir Surveiller et notifier l'activité de l'utilisateur Compte AWS root.

Évaluer la conformité de la MFA de l'utilisateur root

Les services suivants peuvent aider à évaluer la conformité à la MFA pour les informations d'identification de l'utilisateur racine.

> MFA-related les règles renvoient des informations non conformes si vous suivez la meilleure pratique qui consiste à supprimer les informations d'identification de l'utilisateur root.
>
> Nous recommandons de supprimer les informations d'identification de l'utilisateur racine des comptes membres dans votre organisation afin d'empêcher toute utilisation non autorisée. Après avoir supprimé les informations d'identification de l'utilisateur racine, y compris la MFA, ces comptes de membres sont évalués comme non applicables.

- AWS Config fournit des règles pour contrôler la conformité aux meilleures pratiques de l'utilisateur root. Vous pouvez utiliser des règles AWS Config gérées pour vous aider à appliquer l'authentification multifacteur pour les informations d'identification des utilisateurs root. AWS Config peut également identifier les clés d'accès de l'utilisateur root.
- Security Hub CSPM vous fournit une vue complète de votre état de sécurité et vous aide à évaluer votre AWS environnement par rapport aux normes AWS et aux meilleures pratiques du secteur de la sécurité, telles que l'utilisation du MFA sur l'utilisateur root et l'absence de clés d'accès utilisateur root. Pour plus de détails sur les règles disponibles, consultez les Gestion des identités et des accès AWS contrôles dans le guide de l'utilisateur du Security Hub CSPM.

- Trusted Advisor fournit un contrôle de sécurité qui vous permet de savoir si l'authentification multifacteur n'est pas activée sur le compte utilisateur root. Pour de plus amples informations, consultez MFA pour un compte root dans le Guide de l'utilisateur AWS .

Si vous devez signaler un problème de sécurité lié à votre compte, consultez Signaler des e-mails suspects ou Signaler une vulnérabilité. Vous pouvez également contacter AWS pour obtenir de l'aide et des conseils supplémentaires.

Cas d'utilisation métier pour IAM

Un cas d'utilisation professionnel simple de l'IAM peut vous aider à comprendre les principales manières dont vous pouvez implémenter le service afin de contrôler l' AWS accès de vos utilisateurs. Celui-ci est décrit dans des termes généraux, sans les mécanismes d'utilisation de l'API IAM pour obtenir les résultats escomptés.

Ce cas d'utilisation présente deux manières standard dont une entreprise fictive appelée Example Corp peut utiliser IAM. Le premier scénario prend en compte Amazon Elastic Compute Cloud (Amazon EC2). Le second prend en compte Amazon Simple Storage Service (Amazon S3).

Pour plus d'informations sur l'utilisation d'IAM avec d'autres services de AWS, consultezAWS services qui fonctionnent avec IAM.

Rubriques

- Configuration initiale d'Example Corp
- Cas d'utilisation pour IAM avec Amazon EC2
- Cas d'utilisation pour IAM avec Amazon S3

Configuration initiale d'Example Corp

Nikki Wolf et Mateo Jackson sont les fondateurs d'Example Corp. Au démarrage de l'entreprise, ils créent Compte AWS et configurent AWS IAM Identity Center(IAM Identity Center) pour créer des comptes administratifs à utiliser avec leurs ressources. AWS Lorsque vous configurez l'accès au compte pour l'utilisateur administratif, IAM Identity Center crée un rôle IAM correspondant. Ce rôle, qui est contrôlé par IAM Identity Center, est créé dans le rôle approprié Compte AWS et les politiques spécifiées dans l'ensemble d'AdministratorAccessautorisations sont associées au rôle.

Comme ils ont désormais des comptes administrateurs, Nikki et Mateo n'ont plus besoin d'utiliser leur utilisateur root pour accéder à leur Compte AWS. Ils ne prévoient de l'utiliser que pour effectuer les tâches que lui seul peut effectuer. Après avoir passé en revue les bonnes pratiques de sécurité, ils configurent une authentification multifactorielle (MFA) pour leurs informations d'identification d'utilisateur root et décident de la manière de protéger ces informations d'identification d'utilisateur root.

Au fur et à mesure où leur entreprise croît, ils embauchent des employés comme développeurs, administrateurs, testeurs, gestionnaires et administrateurs système. Nikki est en charge des opérations, tandis que Mateo gère les équipes d'ingénierie. Ils ont mis en place un serveur de domaine Active Directory pour gérer les comptes des employés et gérer l'accès aux ressources internes de l'entreprise.

Pour permettre à leurs employés d'accéder aux AWS ressources, ils utilisent IAM Identity Center pour connecter l'Active Directory de leur entreprise au leur Compte AWS.

Comme ils ont connecté Active Directory à IAM Identity Center, les utilisateurs, les groupes et les appartenances aux groupes sont synchronisés et définis. Ils doivent attribuer des ensembles d'autorisations et des rôles aux différents groupes afin de donner aux utilisateurs le niveau d'accès correct aux AWS ressources. Ils utilisent AWS politiques gérées pour les fonctions professionnelles Console de gestion AWS pour créer ces ensembles d'autorisations :

- Administrateur
- Facturation
- Développeurs
- Administrateurs réseau
- Administrateurs de base de données
- Administrateurs système
- Utilisateurs du support

Ils attribuent ensuite ces ensembles d'autorisations aux rôles attribués à leurs groupes Active Directory.

Pour obtenir un guide détaillé décrivant la configuration initiale d'IAM Identity Center, consultez Mise en route dans le Guide de l'utilisateur AWS IAM Identity Center . Pour plus d'informations sur l'approvisionnement de l'accès utilisateur à IAM Identity Center, consultez Accès par authentification unique aux comptes AWS dans le Guide de l'utilisateur AWS IAM Identity Center .

Cas d'utilisation pour IAM avec Amazon EC2

Une entreprise comme Example Corp utilise généralement IAM pour interagir avec des services comme Amazon EC2. Pour comprendre cette partie du cas d'utilisation, vous devez comprendre les bases d'Amazon EC2. Pour plus d'informations sur Amazon EC2, consultez le guide de l'utilisateur Amazon EC2.

Autorisations Amazon EC2 pour les groupes d'utilisateurs

Pour fournir un contrôle « périmétrique », Nikki associe une politique au groupe AllUsers d'utilisateurs. Cette politique refuse toute AWS demande d'un utilisateur si l'adresse IP d'origine se trouve en dehors du réseau d'entreprise d'Example Corp.

Chez Example Corp, différents groupes IAM nécessitent des autorisations distinctes :

- Administrateurs système : nécessitent l'autorisation de créer et de gérer des AMI, des instances, des instantanés, des volumes, des groupes de sécurité, etc. Nikki associe la politique `AmazonEC2FullAccess` AWS gérée au groupe SysAdmins d'utilisateurs qui autorise les membres du groupe à utiliser toutes les actions Amazon EC2.
- Développeurs : ont uniquement besoin de pouvoir utiliser des instances. Par conséquent, Mateo crée et attache une politique au groupe d'utilisateurs Développeurs qui autorise les développeurs à appeler `DescribeInstances`, `RunInstances`, `StopInstances`, `StartInstances` et `TerminateInstances`.

  > **Note**
  >
  > Amazon EC2 utilise des clés SSH, des mots de passe Windows et des groupes de sécurité pour contrôler qui a accès au système d'exploitation d'instances Amazon EC2 spécifiques. Il n'existe aucune méthode dans le système IAM pour autoriser ou refuser l'accès au système d'exploitation d'une instance spécifique.

- Utilisateurs du support : ne doivent pas pouvoir exécuter toutes les actions Amazon EC2, uniquement répertorier les ressources Amazon EC2 disponibles actuellement. Par conséquent, Nikki crée et attache une politique au groupe d'utilisateurs Utilisateurs du support qui les autorise uniquement à appeler des opérations d'API « Describe » Amazon EC2.

Pour obtenir des exemples de ce à quoi ces politiques peuvent ressembler, consultez Exemples de politiques basées sur l'identité IAM la section Utilisation de la gestion des AWS identités et des accès dans le guide de l'utilisateur Amazon EC2.

Modification de la fonction de tâche de l'utilisateur

À un moment donné, l'un des développeurs, Paulo Santos, change de rôle et devient gestionnaire. En tant que gestionnaire, Paulo fait partie du groupe Utilisateurs du support afin de pouvoir ouvrir des dossiers d'assistance pour ses développeurs. Mateo déplace Paulo du groupe Développeurs vers le groupe d'utilisateurs Utilisateurs du support. En conséquence de ce déplacement, sa capacité à interagir avec des instances Amazon EC2 est limitée. Il ne peut pas lancer ou démarrer des instances. Il ne peut également pas arrêter ou mettre hors service les instances existantes, même s'il était l'utilisateur qui a lancé ou démarré l'instance. Il peut uniquement répertorier les instances lancées par les utilisateurs d'Example Corp.

Cas d'utilisation pour IAM avec Amazon S3

Les entreprises comme Example Corp utilisent aussi généralement IAM avec Amazon S3. John a créé un compartiment Amazon S3 appelé amzn-s3-demo-bucket pour l'entreprise.

Création d'autres utilisateurs et groupes d'utilisateurs

En tant qu'employés, Zhang Wei et Mary Major ont besoin de créer chacun leurs propres données dans le compartiment de l'entreprise. Ils ont également besoin des données partagées en lecture et en écriture que tous les développeurs utiliseront. Pour cela, Mateo organise logiquement les données dans amzn-s3-demo-bucket à l'aide d'un schéma de préfixe de clé Amazon S3 présenté dans l'illustration suivante.

```
/amzn-s3-demo-bucket
    /home
        /zhang
        /major
    /share
        /developers
        /managers
```

Mateo divise `/amzn-s3-demo-bucket` en un ensemble de répertoires de base pour chaque employé et une zone partagée de groupes de développeurs et de gestionnaires.

À présent, Mateo crée un ensemble de politiques pour attribuer des autorisations aux utilisateurs et aux groupes d'utilisateurs :

- Accès au répertoire de base de Zhang : Mateo attache une politique à Wei lui permettant de lire, d'écrire et de répertorier les objets avec le préfixe de clé Amazon S3 `/amzn-s3-demo-bucket/home/zhang/`
- Accès au répertoire de base de Major : Mateo attache une politique à Mary lui permettant de lire, d'écrire et de répertorier les objets avec le préfixe de clé Amazon S3 `/amzn-s3-demo-bucket/home/major/`
- Accès au répertoire partagé par le groupe d'utilisateurs Développeurs : Mateo attache une politique au groupe qui permet aux développeurs de lire, d'écrire et de répertorier tous les objets de `/amzn-s3-demo-bucket/share/developers/`
- Accès au répertoire partagé par le groupe d'utilisateurs Gestionnaires : Mateo attache une politique au groupe d'utilisateurs qui permet aux gestionnaires de lire, d'écrire et de répertorier tous les objets de `/amzn-s3-demo-bucket/share/managers/`

> **Note**
>
> Amazon S3 n'attribue pas automatiquement à un utilisateur qui crée un compartiment ou un objet l'autorisation d'exécuter d'autres actions sur ce compartiment ou cet objet. Par conséquent, dans vos politiques IAM, vous devez accorder explicitement l'autorisation aux utilisateurs d'utiliser les ressources Amazon S3 qu'ils créent.

Pour obtenir des exemples de ce à quoi ces politiques peuvent ressembler, veuillez consulter Access Control (contrôle d'accès) dans le guide de l'utilisateur service de stockage simple Amazon. Pour plus d'informations sur la manière dont les politiques sont évaluées au moment de l'exécution, consultez Logique d'évaluation de politiques.

Modification de la fonction de tâche de l'utilisateur

À un moment donné, l'un des développeurs, Zhang Wei, change de rôle et devient gestionnaire. Nous supposerons qu'il n'a plus besoin d'accéder aux documents de l'annuaire `share/developers`. Mateo, en tant qu'administrateur, déplace Wei du groupe d'utilisateurs `Managers` vers le groupe d'utilisateurs `Developers`. En une simple réaffectation, Wei obtient automatiquement toutes les autorisations accordées au groupe d'utilisateurs `Managers`, mais il en peut plus accéder aux données du répertoire `share/developers`.

Intégration d'une entreprise tierce

Les organisations travaillent souvent avec des entreprises partenaires, des consultants et des sous-traitants. Example Corp a un partenaire appelé Widget Company, et une employée de Widget Company appelé Shirley Rodriguez doit placer des données dans un compartiment à l'intention d'Example Corp. Nikki crée un groupe d'utilisateurs appelé WidgetCo et un utilisateur nommé `Shirley` et ajoute Shirley au groupe WidgetCo d'utilisateurs. Nikki crée également un compartiment spécial appelé amzn-s3-demo-bucket1 pour que Shirley puisse l’utiliser.

Nikki met à jour les politiques existantes ou ajoute de nouvelles politiques pour le partenaire Widget Company. Par exemple, Nikki peut créer une nouvelle politique qui refuse aux membres du groupe WidgetCo d'utilisateurs la possibilité d'utiliser des actions autres que l'écriture. Cette politique est nécessaire uniquement s'il existe une vaste politique qui donne à tous les utilisateurs l'accès à un éventail d'actions Amazon S3.

Didacticiels IAM

Les didacticiels suivants présentent end-to-end des procédures complètes pour les tâches courantes pour Gestion des identités et des accès AWS (IAM). Ils sont conçus pour une environnement de travaux pratiques, avec des exemples fictifs de noms de sociétés, de noms d'utilisateur, etc. Leur objectif consiste à fournir des instructions générales. Ils ne sont pas conçus pour être utilisés directement dans votre environnement de production sans être préalablement vérifiés et adaptés soigneusement aux besoins uniques de votre organisation.

Tutoriels

- Tutoriel IAM : accès délégué via AWS comptes utilisant des rôles IAM
- Didacticiel IAM : créer et attacher votre première politique gérée par le client
- Tutoriel IAM : définir les autorisations d'accès AWS ressources basées sur des balises
- Didacticiel IAM : permettre aux utilisateurs de gérer leurs informations d'identification et leurs paramètres MFA
- Tutoriel IAM : utilisez un CloudFormation modèle pour créer un fournisseur d'identité SAML (IdP)
- Tutoriel IAM : utilisez un CloudFormation modèle pour créer un rôle IAM fédéré SAML
- Tutoriel IAM : utilisez un CloudFormation modèle pour créer un fournisseur d'identité SAML (IdP) et un rôle IAM fédéré SAML

Tutoriel IAM : accès délégué via AWS comptes utilisant des rôles IAM

> **Important**
>
> Les meilleures pratiques IAM recommandent de demander aux utilisateurs humains d'utiliser la fédération avec un fournisseur d'identité pour accéder à l'aide d'informations d'identification temporaires au lieu d' AWS utiliser des utilisateurs IAM avec des informations d'identification à long terme. Nous vous recommandons de n'utiliser les utilisateurs IAM que pour les cas d'utilisation spécifiques non pris en charge par les utilisateurs fédérés.

Ce tutoriel vous apprend à utiliser un rôle pour déléguer l'accès à des ressources dans des Comptes AWS différents appelés Destination et Origine. Vous partagez les ressources d'un compte

avec les utilisateurs d'un autre compte. En configurant l'accès entre les comptes de cette manière, vous n'avez pas besoin de créer des utilisateurs IAM individuels dans chaque compte. En outre, les utilisateurs n'ont pas besoin de se déconnecter d'un compte et de se connecter à un autre compte pour accéder à des ressources situées dans des Comptes AWS différents. Après avoir configuré le rôle, vous verrez comment l'utiliser à partir de Console de gestion AWS l' AWS CLI API et de l'API.

Dans ce didacticiel, le compte Destination gère les données d’application auxquelles accèdent différentes applications et équipes. Dans chaque compte, vous stockez les informations de l'application dans des compartiments Amazon S3. Vous gérez les utilisateurs IAM dans le compte Origine,où vous disposez de deux rôles d’utilisateur IAM : Développeurs et Analystes. Les développeurs et les analystes utilisent le compte Origine pour générer des données partagées par plusieurs microservices. Les deux rôles ont des autorisations pour travailler dans le compte Origine et y accéder aux ressources qui s’y trouvent. De temps à autre, un développeur doit mettre à jour les données partagées dans le compte Destination. Les développeurs stockent ces données dans un compartiment Amazon S3 appelé `amzn-s3-demo-bucket-shared-container`.

Au terme de ce tutoriel, vous disposerez des éléments suivants :

- Utilisateurs du compte Origine (le compte approuvé) autorisés à endosser un rôle spécifique dans le compte Destination.
- Rôle du compte Destination (le compte d’approbation) autorisé à accéder à un compartiment Amazon S3 spécifique.
- Le compartiment `amzn-s3-demo-bucket-shared-container` du compte Destination.

Les développeurs peuvent utiliser le rôle dans le Console de gestion AWS pour accéder au `amzn-s3-demo-bucket-shared-container` compartiment dans le compte Destination. Ils peuvent également accéder au compartiment à l'aide des appels API authentifiés par les informations d'identification temporaires fournies par le rôle. Les tentatives similaires des analystes pour utiliser le rôle échouent.

Ce flux de travail se compose de trois étapes de base :

Création d’un rôle dans le compte Destination

Tout d'abord, vous l'utilisez Console de gestion AWS pour établir la confiance entre le compte de destination (numéro d'identification 999999999999) et le compte d'origine (numéro d'identification 111111111111). Vous commencez par créer un rôle IAM nommé UpdateData. Lorsque vous créez le rôle, vous définissez le compte Origine en tant qu’entité approuvée et spécifiez une

politique d'autorisations qui autorise les utilisateurs de confiance à mettre à jour le compartiment `amzn-s3-demo-bucket-shared-container`.

Accorder l'accès au rôle

Dans cette section, vous modifiez la politique de rôle pour refuser aux analystes l'accès au rôle `UpdateData`. Parce que les analystes ont PowerUser accès dans ce scénario et que vous devez explicitement refuser l'utilisation du rôle.

Tester l'accès en changeant de rôles

Enfin, en tant que développeur, vous utilisez le rôle `UpdateData` pour mettre à jour le compartiment `amzn-s3-demo-bucket-shared-container` dans le compte Destination. Vous voyez comment accéder au rôle via la AWS console AWS CLI, l'API et.

Considérations

Avant d'utiliser les rôles IAM pour déléguer l'accès aux ressources Comptes AWS, il est important de prendre en compte les points suivants :

- Il n'est pas possible de passer à un rôle si vous vous connectez en tant qu' Utilisateur racine d'un compte AWS.
- Les politiques IAM basées sur les ressources et les rôles ne délèguent l'accès entre les comptes qu'au sein d'une seule partition. Par exemple, supposons que vous avez un compte dans la région USA Ouest (Californie du Nord) sur la partition `aws` standard. Vous avez également un compte dans la région Chine (Beijing) sur la partition `aws-cn`. Vous ne pouvez pas utiliser une politique basée sur les ressources d'Amazon S3 dans votre compte en Chine (Beijing) pour autoriser l'accès aux utilisateurs de votre compte `aws` standard.
- Vous pouvez l'utiliser AWS IAM Identity Center pour faciliter l'authentification unique (SSO) pour les comptes externes Comptes AWS (comptes extérieurs au vôtre) à l'aide du langage AWS Organizations SAML (Security Assertion Markup Language). Pour plus de détails, voir Intégrer des composants externes Comptes AWSAWS IAM Identity Center pour une gestion centralisée des accès avec une facturation indépendante à l'aide de SAML 2.0
- Vous pouvez associer des rôles à des AWS ressources telles que des instances ou des AWS Lambda fonctions Amazon EC2. Pour en savoir plus, consultez Créez un rôle pour déléguer des autorisations à un AWS web.

- Si vous souhaitez qu'une application joue un rôle dans une autre Compte AWS, vous pouvez utiliser le AWS SDK pour l'attribution de rôles entre comptes. Pour en savoir plus, consultez Authentification et accès dans le Guide de référence des AWS SDK et outils.
- Le changement de rôle à l'aide de the Console de gestion AWS ne fonctionne qu'avec les comptes qui ne nécessitent pas de`ExternalId`. Par exemple, supposons que vous accordiez l'accès à votre compte à un tiers et que vous ayez besoin d'un `ExternalId` dans un élément `Condition` de votre politique d'autorisations. Dans ce cas, le tiers peut accéder à votre compte uniquement à l'aide de l' AWS API ou d'un outil de ligne de commande. Le tiers ne peut pas utiliser la console, car elle doit fournir de valeur pour `ExternalId`. Pour plus d'informations sur ce scénarioAccès à Comptes AWS appartenant à des tiers, consultez et Comment activer l'accès multicompte Console de gestion AWS au blog sur la AWS sécurité.

Conditions préalables

Le didacticiel présume que vous avez déjà ce qui suit en place :

- Deux comptes distincts Comptes AWS que vous pouvez utiliser, l'un pour représenter le compte d'origine et l'autre pour représenter le compte de destination.
- Les utilisateurs et les rôles du compte Origine créés et configurés comme suit :

Titre de la tâche	Utilisateur	Permissions
Developer	David	Les deux utilisateurs peuvent se connecter et utiliser Console de gestion AWS le compte d'origine.
Analyste	Jane	

- Vous n'avez pas besoin de créer d'utilisateurs dans le compte Destination.
- Un compartiment Amazon S3 créé dans le compte Destination. Vous pouvez l'appeler `amzn-s3-demo-bucket-shared-container` dans ce tutoriel, mais du fait que les noms de compartiment S3 doivent être globalement uniques, vous devrez utiliser un compartiment avec un nom différent.

Création d'un rôle dans le compte Destination

Vous pouvez autoriser les utilisateurs de l'un Compte AWS à accéder aux ressources d'un autre Compte AWS. Dans ce didacticiel, nous le ferons en créant un rôle qui définit qui peut y accéder et quelles autorisations il accorde aux utilisateurs qui y basculent.

Dans cette étape du didacticiel, vous créez le rôle dans le compte Destination et spécifiez le compte Origine comme entité approuvée. Vous limitez également les autorisations du rôle à un accès en lecture seule et en écriture au compartiment `amzn-s3-demo-bucket-shared-container`. Toute personne ayant autorisation d'utiliser le rôle peut lire et écrire dans le compartiment `shared-container`.

Avant de pouvoir créer un rôle, vous avez besoin de l'ID de compte de l'utilisateur d'origine Compte AWS. Un identifiant de compte unique est attribué à chacun Compte AWS .

Pour obtenir le code d'origine Compte AWS ID

1. Connectez-vous en Console de gestion AWS tant qu'administrateur du compte d'origine et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans la console IAM, sélectionnez votre nom d'utilisateur dans la barre de navigation, en haut à droite. Cela ressemble généralement à ceci : ***username*** @ ***account_ID_number_or_alias***.

   Pour ce scénario, vous pouvez utiliser l'ID 111111111111 pour le compte Origine. Toutefois, vous devez utiliser un ID de compte valide si vous utilisez ce scénario dans votre environnement de test.

Pour créer un rôle dans le compte Destination qui peut être utilisé par le compte Origine

1. Connectez-vous en Console de gestion AWS tant qu'administrateur du compte Destination et ouvrez la console IAM.
2. Avant de créer le rôle, préparez la politique gérée qui définit les autorisations pour les exigences du rôle. Vous attachez cette politique au rôle dans une étape ultérieure.

   Vous devez définir l'accès en lecture et en écriture au compartiment `amzn-s3-demo-bucket-shared-container`. Bien qu'elle AWS fournisse certaines politiques gérées par Amazon S3, aucune n'offre un accès en lecture et en écriture à un seul compartiment Amazon S3. Vous pouvez créer votre propre politique à la place.

   Dans le panneau de navigation, choisissez Politiques, puis Créer une politique.
3. Choisissez l'onglet JSON et copiez le texte du document de politique JSON suivant. Collez ce texte dans la zone de texte JSON, en remplaçant l'ARN de la ressource (`arn:aws:s3:::shared-container`) par celui qui correspond véritablement à votre compartiment Amazon S3.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:GetBucketLocation"
       ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-shared-container"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-shared-container/*"
    }
  ]
}
```

L'action `ListAllMyBuckets` accorde l'autorisation de répertorier tous les compartiments appartenant à l'expéditeur authentifié de la demande. L'autorisation `ListBucket` permet aux utilisateurs d'afficher des objets dans le compartiment `amzn-s3-demo-bucket-shared-container`. Les autorisations `GetObject`, `PutObject`, `DeleteObject` permettent aux utilisateurs d'afficher, de mettre à jour et de supprimer le contenu du compartiment `amzn-s3-demo-bucket-shared-container`.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour de plus amples informations, veuillez consulter Restructuration de politique.

4. Sur la page Vérifier et créer, tapez **`read-write-app-bucket`** pour le nom de la politique. Vérifiez les autorisations accordées par votre politique, puis choisissez Créer une politique pour enregistrer votre travail.

   La nouvelle politique apparaît dans la liste des politiques gérées.

5. Dans le panneau de navigation, choisissez Rôles, puis Créer un rôle.
6. Choisissez le type de rôle Un Compte AWS.
7. Dans le champ ID de compte, saisissez l'ID du compte Origine.

   Ce didacticiel utilise l'exemple d'ID de compte **`111111111111`** pour le compte Origine. Vous devez utiliser un ID de compte valide. Si vous utilisez un ID de compte non valide, comme **`111111111111`**, IAM ne vous laisse pas créer de rôle.

   Pour le moment, les utilisateurs n'ont pas besoin d'un ID externe ou d'une authentification multi-facteurs (MFA) pour endosser le rôle. Laissez ces options décochées. Pour plus d'informations, veuillez consulter AWS Multi-factor authentification dans IAM.

8. Choisissez suivant : autorisations pour configurer les autorisations associées au rôle.
9. Cochez la case en regard de la politique que vous avez créée précédemment.

> **Conseil**
>
> Pour filtre, sélectionnez politiques gérées par le client pour affiner la liste afin d'inclure uniquement les politiques que vous avez créées. Cela masque les politiques créées par AWS et permet de rechercher plus facilement celle dont vous avez besoin.

Ensuite, choisissez Suivant.

10. (Facultatif) Ajoutez des métadonnées au rôle en associant les identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Tags pour Gestion des identités et des accès AWS resources.
11. (Facultatif) Pour Description, saisissez une description pour le nouveau rôle.
12. Après avoir procédé à la révision du rôle, sélectionnez Créer un rôle.

    Le rôle `UpdateData` s'affiche dans la liste des rôles.

À présent, vous devez obtenir l'Amazon Resource Name (ARN) du rôle, qui est un identifiant unique du rôle. Lorsque vous modifiez le rôle du développeur dans le compte Origine, vous spécifiez l’ARN du rôle à partir du compte Destination afin d’accorder ou de refuser des autorisations.

Pour obtenir l'ARN pour UpdateData

1. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles).
2. Dans la liste des rôles, sélectionnez le rôle `UpdateData`.
3. Dans la section Récapitulatif du panneau des détails, copiez la valeur du champ ARN de rôle.

   Le compte Destination ayant pour ID de compte 999999999999, l’ARN du rôle est donc `arn:aws:iam::999999999999:role/UpdateData`. Assurez-vous de fournir le véritable Compte AWS identifiant du compte Destination.

À ce stade, vous avez établi un lien d’approbation entre les comptes Destination et Origine. Pour ce faire, créez un rôle dans le compte Destination qui identifie le compte Origine en tant que principal compte d’approbation. Vous avez également défini ce que les utilisateurs qui basculent vers le rôle `UpdateData` peuvent faire.

Ensuite, modifiez les autorisations pour le rôle Développeur.

Accorder l'accès au rôle

À ce stade, les analystes et les développeurs disposent d’autorisations leur permettant de gérer les données dans le compte Origine. Utilisez la procédure requise pour ajouter des autorisations visant à autoriser le changement de rôle.

Pour modifier le rôle des développeurs afin de leur permettre de passer au UpdateData rôle

1. Connectez-vous en tant qu'administrateur dans le compte Origine, puis ouvrez la console IAM.
2. Choisissez Rôles, puis choisissez Développeurs.
3. Sélectionnez l'onglet Permissions (Autorisations), puis Add permissions (Ajouter des autorisations), et enfin Create inline policy (Créer une politique en ligne).
4. Choisissez l'onglet JSON.
5. Ajoutez l'instruction de politique suivante pour autoriser l'action `AssumeRole` sur le rôle `UpdateData` dans le compte Destination. Assurez-vous de remplacer *`DESTINATION-ACCOUNT-ID`* l'`Resource`élément par l' Compte AWS identifiant réel du compte de destination.

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": {
           "Effect": "Allow",
           "Action": "sts:AssumeRole",
           "Resource": "arn:aws:iam::111122223333:role/UpdateData"
       }
   }
   ```

   L'effet `Allow` autorise explicitement l'accès du groupe Développeurs au rôle `UpdateData` dans le compte Destination. Tout développeur qui essaie d'accéder à ce rôle y parvient.
6. Choisissez Examiner une politique.
7. Tapez un Nom, tel que **`allow-assume-S3-role-in-destination`**.
8. Choisissez Create Policy (Créer une politique).

Dans la plupart des environnements, vous n'aurez peut-être pas besoin de la procédure suivante. Si, toutefois, vous utilisez PowerUserAccess des autorisations, il se peut que certains groupes soient déjà en mesure de changer de rôle. Les procédures suivantes indiquent comment ajouter une autorisation `"Deny"` au groupe Analystes pour s'assurer qu'ils ne peuvent pas endosser le rôle. Si vous n'avez pas besoin de cette procédure dans votre environnement, nous vous recommandons de ne pas l'ajouter. Les autorisations `"Deny"` rendent l'ensemble des autorisations plus compliqué à

gérer et à comprendre. Utilisez les autorisations "Deny" uniquement lorsque vous ne disposez pas de meilleure option.

Pour modifier le rôle Analystes afin de lui refuser l'autorisation d'endosser ce rôle **UpdateData**

1. Choisissez Rôles, puis Analystes.
2. Sélectionnez l'onglet Permissions (Autorisations), puis Add permissions (Ajouter des autorisations), et enfin Create inline policy (Créer une politique en ligne).
3. Choisissez l'onglet JSON.
4. Ajoutez l'instruction de politique suivante pour refuser l'action `AssumeRole` sur le rôle `UpdateData`. Assurez-vous de remplacer *DESTINATION-ACCOUNT-ID* l'`Resource`élément par l' Compte AWS identifiant réel du compte de destination.

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": {
           "Effect": "Deny",
           "Action": "sts:AssumeRole",
           "Resource": "arn:aws:iam::111122223333:role/UpdateData"
       }
   }
   ```

   L'effet `Deny` refuse explicitement l'accès du groupe Analystes au rôle `UpdateData` dans le compte Destination. Tout analyste qui essaiera d'accéder au rôle recevra un message d'accès refusé.
5. Choisissez Examiner une politique.
6. Tapez un Nom comme **deny-assume-S3-role-in-destination**.
7. Choisissez Create Policy (Créer une politique).

Le rôle Développeurs a maintenant l'autorisation d'utiliser le rôle `UpdateData` dans le compte Destination. Le rôle Analystes ne peut pas utiliser le rôle `UpdateData`.

Ensuite, vous pouvez voir comment David, un développeur, peut accéder au compartiment `amzn-s3-demo-bucket-shared-container` dans le compte Destination. David peut accéder au bucket depuis la Console de gestion AWS AWS CLI, l'API ou l' AWS API.

Tester l'accès en changeant de rôles

Après avoir terminé les deux premières étapes de ce didacticiel, vous disposez d'un rôle qui accorde l'accès à une ressource du compte Destination. Vous disposez également d'un rôle dans le compte Origine avec des utilisateurs autorisés à utiliser ce rôle. Cette étape explique comment tester le passage à ce rôle à partir de Console de gestion AWS l' AWS CLI API et de l' AWS API.

Pour obtenir de l'aide sur les problèmes courants que vous êtes susceptible de rencontrer lors de l'utilisation des rôles IAM, consultez Résolution des problèmes liés aux rôles IAM.

Changer de rôle (console)

Si David a besoin de mettre à jour les données du compte Destination dans le Console de gestion AWS, il peut le faire en utilisant Switch Role. Il indique l'ID de compte ou l'alias et le nom du rôle, et ses autorisations passent immédiatement à celles autorisées par le rôle. Il peut ensuite utiliser la console pour travailler avec le compartiment `amzn-s3-demo-bucket-shared-container`, mais il ne peut pas utiliser d'autres ressources de l'environnement Destination. Bien que David utilise le rôle, il ne peut pas non plus utiliser ses privilèges d'utilisateur avancé dans le compte Origine. Ceci est dû au fait qu'un seul ensemble d'autorisations peut être actif à la fois.

IAM fournit deux procédures que David peut utiliser pour accéder à la page Switch Role (changer de rôle) :

- David reçoit un lien de son administrateur qui dirige vers la configuration Switch Role prédéfinie. Le lien est fourni à l'administrateur sur la page finale de l'assistant Créer un rôle ou sur la page Résumé du rôle pour un rôle inter-compte. Si David choisit ce lien, il accède à la page Switch Role (Changer de rôle) avec les champs ID de compte et Nom du rôle déjà remplis. Il ne reste plus à David qu'à choisir Changer de rôle.
- L'administrateur n'envoie pas le lien dans un e-mail, mais il envoie les valeurs de l'ID de compte et du Nom de rôle. Pour changer de rôle, David doit manuellement saisir les valeurs. La procédure suivante en est l'illustration.

Pour endosser un rôle

1. David se connecte en Console de gestion AWS utilisant son utilisateur normal dans le compte d'origine.
2. Il choisit le lien que l'administrateur lui a envoyé par email. Cela amène David à la page Switch Role (Changer de rôle) avec l'identifiant ou l'alias de compte et les informations sur le nom du rôle déjà remplis.

   —ou—

   David choisit son nom (le menu Identity [Identité]) dans la barre de navigation, puis sélectionne Switch Roles (Changer de rôle).

   Si c'est la première fois que David essaie d'accéder à la page Switch Role (changer de rôle) de cette manière, il est dirigé vers la 1ère page de mise en route Switch Role (changer de rôle). Cette page fournit des informations supplémentaires sur la manière dont le changement de rôle permet aux utilisateurs de gérer des ressources entre Comptes AWS. David doit choisir Switch Role (changer de rôle) sur cette page pour appliquer le reste de la procédure.
3. Ensuite, pour accéder au rôle, David doit saisir manuellement le numéro d’ID (999999999999) et le nom du rôle (`UpdateData`) du compte Destination.

   En outre, David souhaite contrôler les rôles et les autorisations associées actuellement actifs dans IAM. Pour suivre ces informations, il saisit `Destination` dans la zone de texte Display Name (nom complet), choisit l'option de couleur rouge, puis choisit Switch Role (changer de rôle).
4. David peut maintenant utiliser la console Amazon S3 pour travailler avec le compartiment Amazon S3 ou toute autre ressource pour laquelle le rôle `UpdateData` dispose d'autorisations.
5. Quand c'est fait, David peut retourner dans ses autorisations d'origine. Pour cela, il choisit le nom complet du rôle Destination sur la barre de navigation, puis il sélectionne Revenir à David à 111111111111.
6. La prochaine fois que David voudra changer de rôle et choisira le menu Identité dans la barre de navigation, il verra l’entrée Destination toujours affichée depuis la dernière fois. Il lui suffira de choisir cette entrée pour changer de rôle immédiatement sans avoir à ressaisir l'ID du compte et le nom du rôle.

Changer de rôle (AWS CLI)

Si David a besoin de travailler dans l'environnement Destination dans la ligne de commande, il peut y parvenir grâce à l'outil AWS CLI. Il exécute la commande `aws sts assume-role` et transmet l'ARN du rôle pour obtenir les informations d'identification de sécurité temporaires de ce rôle. Il configure ensuite ces informations d'identification dans des variables d'environnement afin que AWS CLI les commandes suivantes fonctionnent en utilisant les autorisations du rôle. Bien que David utilise le rôle, il ne peut pas utiliser ses privilèges d'utilisateur avancé dans le compte Origine, car un seul ensemble d'autorisations peut être effectif à la fois.

Notez que toutes les clés d'accès et tous les jetons sont des exemples uniquement et ne peuvent pas être utilisés comme indiqué. Remplacez-les par les valeurs correspondantes de votre environnement en direct.

Pour endosser un rôle

1. David ouvre une fenêtre d'invite de commande et confirme que le AWS CLI client fonctionne en exécutant la commande suivante :

   ```
   aws help
   ```

   > **Note**
   >
   > L'environnement par défaut de David utilise les informations d'identification de l'utilisateur `David` de son profil par défaut qu'il a créé avec la commande `aws configure`. Pour plus d'informations, veuillez consulter configuration de l'outil AWS Command Line Interface dans le guide de l'utilisateur de l'outil AWS Command Line Interface .

2. Il commence le processus de changement de rôle en exécutant la commande suivante pour passer au rôle `UpdateData` dans le compte Destination. Il a reçu l'ARN du rôle auprès de l'administrateur ayant créé le rôle. La commande nécessite que vous fournissiez un nom de session également. Pour cela, vous pouvez choisir n'importe quel texte.

   ```
   aws sts assume-role --role-arn "arn:aws:iam::999999999999:role/UpdateData" --role-
   session-name "David-ProdUpdate"
   ```

   David voit ensuite ce qui suit dans le résultat :

```
{
    "Credentials": {
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "SessionToken": "AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLE
CvSRyh0FW7jEXAMPLEW+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
uZEXAMPLECihzFB5lTYLto9dyBgSDy
EXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPyOj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3Uuysg
sKdEXAMPLE1TVastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP
+4eZScEXAMPLEsnf87e
NhyDHq6ikBQ==",
        "Expiration": "2014-12-11T23:08:07Z",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
}
```

3. David voit les trois éléments dont il a besoin dans la section informations d'identification du résultat.

   - `AccessKeyId`
   - `SecretAccessKey`
   - `SessionToken`

   David doit configurer l' AWS CLI environnement pour utiliser ces paramètres lors des appels suivants. Pour plus d'informations sur les différentes manières de configurer vos informations d'identification, veuillez consulter Configuration de l'outil AWS Command Line Interface. Vous ne pouvez pas utiliser la commande `aws configure`, car elle ne prend pas en charge la capture du jeton de session. En revanche, vous pouvez saisir manuellement les informations dans un fichier de configuration. Du fait qu'il s'agisse d'informations d'identification temporaires avec un délai d'expiration relativement court, il est plus facile de les ajouter à l'environnement de votre session de ligne de commande actuelle.

4. Pour ajouter les trois valeurs à l'environnement, David coupe et colle le résultat de l'étape précédente dans les commandes suivantes. Vous pourriez vouloir couper et coller dans un simple éditeur de texte pour résoudre les problèmes de retour à la ligne dans le résultat du jeton de session. Elles doivent être ajoutées sous la forme d'une simple chaîne longue, même si elles s'affichent avec des retours de ligne pour plus de clarté.

L'exemple suivant illustre les commandes fournies dans l'environnement Windows où « set » est la commande destinée à créer une variable d'environnement. Sur Linux ou macOS, vous devez plutôt utiliser la commande « export ». Toutes les autres sections de l'exemple sont valides dans les trois environnements.

Pour en savoir plus sur l'utilisation des outils pour Windows Powershell, consultez Passer à un rôle IAM (Outils pour Windows PowerShell)

```
set AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
set AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
set AWS_SESSION_TOKEN=AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLECvS
Ryh0FW7jEXAMPLEW+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
uZEXAMPLECihzFB5lTYLto9dyBgSDyEXA
MPLEKEY9/
g7QRUhZp4bqbEXAMPLENwGPyOj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UusKd
EXAMPLE1TVastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP
+4eZScEXAMPLENhykxiHen
DHq6ikBQ==
```

À ce stade, toutes les commandes seront exécutées avec les autorisations du rôle identifié par ces informations d'identification. Dans le cas de David, le rôle `UpdateData`.

> **Important**
>
> Vous pouvez enregistrer vos paramètres de configuration utilisés fréquemment et vos informations d'identification dans des fichiers qui sont gérés par l' AWS CLI. Pour plus d'informations, consultez Utilisations de fichiers de configurations et d'informations d'identification existants dans le Guide de l'utilisateur AWS Command Line Interface .

5. Exécutez la commande pour accéder aux ressources du compte Destination. Dans cet exemple, David répertorie simplement le contenu de son compartiment S3 avec la commande suivante.

```
aws s3 ls s3://shared-container
```

Du fait que les noms de compartiments Amazon S3 sont universellement uniques, il n'est pas nécessaire de spécifier l'ID du compte qui est titulaire du compartiment. Pour accéder aux

ressources d'autres AWS services, reportez-vous à la AWS CLI documentation de ce service pour connaître les commandes et la syntaxe requises pour référencer ses ressources.

En utilisant AssumeRole (AWS API)

Lorsque David a besoin de faire une mise à jour dans le compte Destination depuis le code, il réalise un appel `AssumeRole` pour endosser le rôle `UpdateData`. L'appel renvoie des informations d'identification temporaires qu'il peut utiliser pour accéder au compartiment `amzn-s3-demo-bucket-shared-container` dans le compte Destination. Grâce à ces informations d'identification, David peut réaliser des appels d'API pour mettre à jour le compartiment `amzn-s3-demo-bucket-shared-container`. Cependant, il ne peut pas réaliser des appels d'API pour accéder aux autres ressources du compte Destination, même s'il a des autorisations d'utilisateur avancé dans le compte Origine.

Pour endosser un rôle

1. David appelle `AssumeRole` dans le cadre d'une application. Il doit spécifier l'ARN `UpdateData` : `arn:aws:iam::999999999999:role/UpdateData`.

   La réponse de l'appel `AssumeRole` inclut les informations d'identification temporaires avec un `AccessKeyId` et un `SecretAccessKey`. Elle inclut également une heure `Expiration` qui indique à quel moment les informations d'identification expirent et vous devez en demander de nouvelles. Lorsque vous configurez le chaînage de rôles avec le AWS SDK, de nombreux fournisseurs d'informations d'identification actualisent automatiquement les informations d'identification avant leur expiration.
2. Grâce aux informations d'identification de sécurité temporaires, David réalise un appel `s3:PutObject` pour mettre à jour le compartiment `amzn-s3-demo-bucket-shared-container`. Il transfère les informations d'identification à l'appel d'API en tant que paramètre `AuthParams`. Du fait que les informations d'identification de rôle temporaires ont uniquement accès en lecture et en écriture au compartiment `amzn-s3-demo-bucket-shared-container`, toute autre action dans le compte Destination est refusée.

Pour obtenir un exemple de code (à l'aide de Python), veuillez consulter Passez à un rôle IAM (AWS API).

Ressources supplémentaires

Les ressources suivantes peuvent vous aider à en savoir plus sur les sujets abordés dans ce didacticiel :

- Pour plus d'informations sur les utilisateurs IAM, consultez IAM identités.
- Pour plus d'informations sur les compartiments Amazon S3, veuillez consulter créer un compartiment dans le Amazon Simple Storage Service User Guide (guide de l'utilisateur du service de stockage simple de Amazon).
- Pour savoir si les principaux des comptes situés en dehors de votre zone de confiance (organisation ou compte de confiance) ont accès à vos rôles, veuillez consulter Qu'est-ce que l'Analyseur d'accès IAM ?.

Résumé

Vous avez terminé le didacticiel d'accès aux API entre les comptes. Vous avez créé un rôle pour établir des relations de confiance avec un autre compte et défini les actions que les entités approuvées peuvent effectuer. Ensuite, vous avez modifié une politique de rôle pour contrôler les utilisateurs IAM qui peuvent accéder au rôle. Par conséquent, les développeurs du compte Origine peuvent faire des mises à jour dans le compartiment `amzn-s3-demo-bucket-shared-container` du compte Destination à l'aide d'informations d'identification temporaires.

Didacticiel IAM : créer et attacher votre première politique gérée par le client

Dans ce didacticiel, vous allez utiliser le Console de gestion AWS pour créer une politique gérée par le client, puis associer cette politique à un utilisateur IAM de votre Compte AWS. La politique que vous créez permet à un utilisateur de test IAM de se connecter directement au Console de gestion AWS avec des autorisations en lecture seule.

Ce flux de travail se compose de trois étapes de base :

Étape 1 : Créer la politique

> Par défaut, les utilisateurs IAM ne sont autorisés à rien faire. Ils ne peuvent pas accéder à la Console de gestion AWS ou à gérer les données qu'elle contient sans votre autorisation. Dans

cette étape, vous créez une politique gérée par le client qui autorise tous les utilisateurs attachés à se connecter à la console.

Étape 2 : Attacher la politique

Lorsque vous attachez une politique à un utilisateur, celui-ci hérite des autorisations d'accès associées à cette politique. Dans cette étape, vous attachez la nouvelle politique à un utilisateur test.

Étape 3 : Tester l'accès utilisateur

Une fois la politique attachée, vous pouvez vous connecter en tant que l'utilisateur pour tester la politique.

Conditions préalables

Pour exécuter les étapes de ce didacticiel, vous devez déjà disposer des éléments suivants :

- Et Compte AWS auquel vous pouvez vous connecter en tant qu'utilisateur IAM avec des autorisations administratives.
- Un utilisateur IAM de test n'a aucune autorisation attribuée et n'appartient à aucun groupe comme suit :

Nom utilisateur	Groupe	Permissions
PolicyUser	<aucune>	<aucune>

Étape 1 : Créer la politique

Au cours de cette étape, vous créez une politique gérée par le client qui permet à tout utilisateur attaché de se connecter Console de gestion AWS avec un accès en lecture seule aux données IAM.

Pour créer la politique de votre utilisateur de test

1. Connectez-vous à la console IAM à https://console.aws.amazon.com/iam/l'adresse de votre utilisateur disposant d'autorisations d'administrateur.
2. Dans le panneau de navigation, sélectionnez Policies (Politiques).
3. Dans le panneau de contenu, sélectionnez Créer une politique.

4. Choisissez l'option JSON et copiez le texte du document de politique JSON suivant. Collez ce texte dans la zone de texte JSON.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [ {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateCredentialReport",
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    } ]
}
```

5. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant.

   > **Note**
   >
   > Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou sélectionnez Examiner une politique dans l'onglet Editeur visuel, IAM peut restructurer votre politique pour optimiser son affichage dans l'éditeur visuel. Pour de plus amples informations, veuillez consulter Restructuration de politique.

6. Sur la page Vérifier et créer, tapez **`UsersReadOnlyAccessToIAMConsole`** pour le nom de la politique. Vérifiez les autorisations accordées par votre politique, puis choisissez Créer une politique pour enregistrer votre travail.

   La nouvelle politique s'affiche dans la liste des politiques gérées et est prête à être attachée.

Étape 2 : Attacher la politique

Ensuite, vous attachez la politique que vous venez de créer à votre utilisateur IAM de test.

Pour attacher la politique à votre utilisateur de test

1. Dans le panneau de navigation de la console IAM, sélectionnez Policies (Politiques).
2. En haut de la liste des politiques, dans la zone de recherche, commencez à taper **`UsersReadOnlyAccessToIAMConsole`** jusqu'à ce que vous puissiez voir votre politique. Cliquez ensuite sur le bouton radio situé à côté UsersReadOnlyAccessToIAMConsoledans la liste.
3. Cliquez sur le bouton Actions, puis choisissez Attach (Attacher).
4. Dans les entités IAM, choisissez l'option de filtrage pour les Utilisateurs.
5. Dans la zone de recherche, commencez à taper **`PolicyUser`** jusqu'à ce que cet utilisateur soit visible dans la liste. Cochez ensuite la case en regard de cet utilisateur dans la liste.
6. Choisissez Attach policy (Attacher la politique).

Vous avez attaché la politique à votre utilisateur de test IAM, ce qui signifie que l'utilisateur dispose maintenant d'un accès en lecture seule à la console IAM.

Étape 3 : Tester l'accès utilisateur

Dans le cadre de ce didacticiel, nous vous recommandons de tester l'accès en vous connectant en tant que chacun des utilisateurs de test afin de vous rendre compte de l'expérience que vivent vos utilisateurs.

Tester l'accès en vous connectant avec votre utilisateur test

1. Connectez-vous à la console IAM à l'adresse https://console.aws.amazon.com/iam/avec votre utilisateur de `PolicyUser` test.
2. Parcourez les pages de la console et essayez de créer un nouvel utilisateur ou un nouveau groupe. Notez que `PolicyUser` peut afficher des données mais ne peut ni créer, ni modifier des données IAM existantes.

Ressources connexes

Pour plus d'informations, consultez les ressources suivantes :

- Politiques gérées et politiques en ligne
- Contrôlez l'accès des utilisateurs IAM au Console de gestion AWS

Résumé

Vous avez terminé toutes les étapes nécessaires pour créer et attacher une politique gérée par l'utilisateur. Par conséquent, vous pouvez vous connecter à la console IAM avec votre compte de test pour voir à quoi ressemble l'expérience de vos utilisateurs.

Tutoriel IAM : définir les autorisations d'accès AWS ressources basées sur des balises

Attribute-based le contrôle d'accès (ABAC) est une stratégie d'autorisation qui définit les autorisations en fonction d'attributs. Dans AWS, ces attributs sont appelés balises. Vous pouvez associer des balises aux ressources IAM, y compris aux entités IAM (utilisateurs ou rôles) et aux AWS ressources. Vous pouvez définir des politiques qui utilisent des clés de condition de balise pour accorder des autorisations à vos principaux en fonction de leurs balises. Lorsque vous utilisez des balises pour contrôler l'accès à vos AWS ressources, vous permettez à vos équipes et à vos ressources de se développer en modifiant moins les AWS politiques. Les politiques ABAC sont plus flexibles que AWS les politiques traditionnelles, qui vous obligent à répertorier chaque ressource individuelle. Pour de plus amples informations sur l'ABAC et ses avantages par rapport aux politiques traditionnelles, veuillez consulter Définition des autorisations en fonction des attributs avec autorisation ABAC.

> **Note**
>
> Vous devez transmettre une valeur unique pour chaque balise de session. AWS Security Token Service ne prend pas en charge les balises de session à valeurs multiples.

Rubriques

- Aperçu du didacticiel
- Conditions préalables
- Étape 1 : Créer des utilisateurs test
- Étape 2 : Créer la politique ABAC
- Étape 3 : Créer les rôles
- Étape 4 : Tester la création de secrets
- Étape 5 : Tester l'affichage des secrets
- Étape 6 : Tester l'adaptabilité

- Étape 7 : Tester la mise à jour et la suppression des secrets
- Résumé
- Ressources connexes
- Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC

Aperçu du didacticiel

Ce didacticiel explique comment créer et tester une politique qui autorise des rôles IAM avec des balises de principal à accéder aux ressources dont les balises correspondent. Lorsqu'un principal fait une demande à AWS, les autorisations lui sont accordées si sa balise correspond à celle des ressources. Cette stratégie permet aux utilisateurs de consulter ou de modifier uniquement les AWS ressources nécessaires à leur travail.

Scénario

Supposons que vous soyez un développeur principal dans une grande entreprise nommée Exemple, ainsi qu'un administrateur IAM expérimenté. Vous savez créer et gérer des utilisateurs, des rôles et des politiques IAM. Vous voulez vous assurer que vos ingénieurs en développement et les membres de l'équipe d'assurance qualité peuvent accéder aux ressources dont ils ont besoin. Vous avez également besoin d'une stratégie qui évolue en même temps que votre entreprise.

Vous choisissez d'utiliser AWS des balises de ressources et des balises principales de rôle IAM pour mettre en œuvre une stratégie ABAC pour les services qui la prennent en charge, en commençant par. AWS Secrets Manager Pour connaître les services prenant en charge l'autorisation basée sur des balises, veuillez consulter AWS services qui fonctionnent avec IAM. Pour savoir quelles clés de condition de balisage vous pouvez utiliser dans une politique avec les actions et les ressources de chaque service, voir Actions, ressources et clés de condition pour les AWS services. Vous pouvez configurer vous-même SAML-based ou votre fournisseur d'identité Web pour qu'il transmette des balises de session à AWS. Lorsque vos employés se fédérent en AWS, leurs attributs sont appliqués au principal entrant qui en AWS résulte. Vous pouvez ensuite utiliser l'ABAC pour autoriser ou refuser des autorisations sur la base de ces attributs. Pour savoir comment l'utilisation de balises de session avec une identité fédérée SAML diffère de ce didacticiel, veuillez consulter Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC.

Les membres de votre équipe d'ingénierie et d'assurance qualité sont sur le projet Pegasus ou le projet Unicorn. Vous sélectionnez les valeurs de balise de 3 caractères ci-dessous pour les projets et l'équipe :

- `access-project` = `peg` pour le projet Pegasus
- `access-project` = `uni` pour le projet Unicorn
- `access-team` = `eng` pour l'équipe d'ingénierie
- `access-team` = `qas` pour l'équipe d'assurance qualité

En outre, vous choisissez d'exiger la balise de répartition des `cost-center` coûts pour activer les rapports AWS de facturation personnalisés. Pour plus d'informations, consultez Utilisation des balises d'allocation des coûts dans le Guide de l'utilisateur AWS Billing and Cost Management .

Résumé des décisions clés

- Les employés se connectent avec leurs informations d'identification utilisateur IAM, puis endossent le rôle IAM pour leur équipe et leur projet. Si votre entreprise possède son propre système d'identité, vous pouvez configurer une fédération pour autoriser les employés à endosser un rôle sans utilisateurs IAM. Pour plus d'informations, veuillez consulter Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC.
- La même politique est attachée à tous les rôles. Les actions sont autorisées ou refusées en fonction des balises.
- Les employés peuvent créer des ressources, mais uniquement s'ils attachent à la ressource les balises qui sont appliquées à leur rôle. Ils peuvent ainsi afficher la ressource après l'avoir créée. Les administrateurs ne sont plus tenus de mettre à jour les politiques avec l'ARN des nouvelles ressources.
- Les employés peuvent lire les ressources appartenant à leur équipe, quel que soit le projet.
- Les employés peuvent mettre à jour et supprimer des ressources appartenant à leur équipe et à leur projet.
- Les administrateurs IAM peuvent ajouter un nouveau rôle pour les nouveaux projets. Ils peuvent créer et baliser un nouvel utilisateur IAM pour lui permettre d'accéder au rôle approprié. Les administrateurs ne sont pas tenus de modifier une politique pour prendre en charge un nouveau projet ou membre de l'équipe.

Dans ce didacticiel, vous allez baliser chaque ressource, baliser les rôles de votre projet et ajouter des politiques aux rôles afin d'autoriser le comportement décrit précédemment. La politique qui en résulte autorise les rôles `Create`, `Read`, `Update` et `Delete` à accéder aux ressources qui ont les mêmes balises que le projet et l'équipe. La politique autorise également l'accès `Read` interprojet pour les ressources qui ont les mêmes balises que l'équipe.

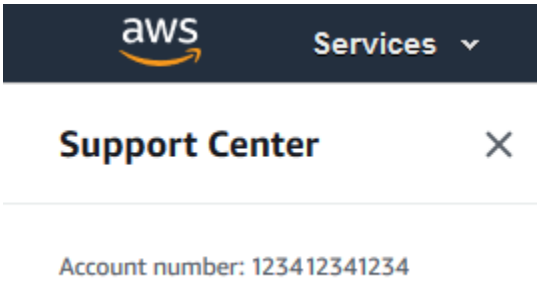
Le diagramme montre deux projets dans lesquels les rôles sont limités à un accès en lecture seule en dehors de leur projet, alors qu'ils ont des autorisations de création, de lecture, de mise à jour et de suppression des ressources dans leur propre projet.

Conditions préalables

Pour exécuter les étapes de ce didacticiel, vous devez déjà disposer des éléments suivants :

- Et Compte AWS auquel vous pouvez vous connecter en tant qu'utilisateur avec des autorisations administratives.
- Votre ID de compte à 12 chiffres, que vous utilisez pour créer les rôles à l'étape 3.

  Pour trouver le numéro d'identification de votre AWS compte à l'aide de Console de gestion AWS, choisissez Support dans la barre de navigation en haut à droite, puis choisissez Centre de support. Le numéro de compte (ID) apparaît dans le panneau de navigation à gauche.



- Une expérience dans la création et la modification d'utilisateurs, de rôles et de politiques IAM dans Console de gestion AWS. Toutefois, si vous avez besoin d'aide pour mémoriser un processus de gestion IAM, le didacticiel fournit des liens vers les instructions étape par étape.

Étape 1 : Créer des utilisateurs test

Pour le test, créez quatre utilisateurs IAM autorisés à endosser des rôles avec les mêmes balises. Cela facilite l'ajout d'utilisateurs à vos équipes. Lorsque vous balisez les utilisateurs, ces derniers bénéficient d'un automatiquement d'un accès pour endosser le rôle. Vous n'avez pas à ajouter les utilisateurs à la politique d'approbation du rôle s'ils travaillent sur un seul projet et dans une seule équipe.

1. Créez la politique gérée par le client ci-dessous et nommez-la `access-assume-role`. Pour de plus amples informations sur la création d'une politique JSON, veuillez consulter Création de politiques IAM.

Politique ABAC : endosser n'importe quel rôle de la politique ABAC, mais uniquement lorsque les balises de l'utilisateur et du rôle correspondent

La politique suivante autorise un utilisateur à endosser n'importe quel rôle de votre compte avec le préfixe de nom `access-`. Le rôle doit également être balisé avec les mêmes balises de projet, d'équipe et de centre de coûts que l'utilisateur.

Pour utiliser cette politique, remplacez le texte en italique de l'espace réservé dans l'exemple de politique par vos propres informations de compte.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TutorialAssumeRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::111122223333:role/access-*",
            "Condition": {
                "StringEquals": {
                    "iam:ResourceTag/access-project": "${aws:PrincipalTag/
access-project}",
                    "iam:ResourceTag/access-team": "${aws:PrincipalTag/
access-team}",
                    "iam:ResourceTag/cost-center": "${aws:PrincipalTag/cost-
center}"
                }
            }
        }
    ]
}
```

Pour étendre ce didacticiel à un plus grand nombre d'utilisateurs, vous pouvez attacher la politique à un groupe et ajouter chaque utilisateur au groupe. Pour plus d'informations, consultez Création de groupes IAM et Modification des utilisateurs dans les groupes IAM.

2. Créez les utilisateurs IAM suivants, attachez la politique d'autorisation `access-assume-role`. Assurez-vous de sélectionner Fournir un accès utilisateur à la Console de gestion AWS, puis d'ajouter les balises suivantes.

Nom utilisateur	Clé de balise utilisateur	Valeur de balise utilisateur
accès- Arnav-peg-eng	`access-project`	`peg`
	`access-team`	`eng`
	`cost-center`	`987654`
accès- Mary-peg-qas	`access-project`	`peg`
	`access-team`	`qas`
	`cost-center`	`987654`
accès- Saanvi-uni-eng	`access-project`	`uni`
	`access-team`	`eng`
	`cost-center`	`123456`
accès- Carlos-uni-qas	`access-project`	`uni`
	`access-team`	`qas`
	`cost-center`	`123456`

Étape 2 : Créer la politique ABAC

Créez la politique suivante et nommez-la **`access-same-project-team`**. Vous allez ajouter cette politique aux rôles ultérieurement. Pour de plus amples informations sur la création d'une politique JSON, veuillez consulter Création de politiques IAM.

Pour accéder à des politiques supplémentaires que vous pouvez adapter pour ce didacticiel, veuillez consulter les pages suivantes :

- Contrôle de l'accès pour les principaux IAM
- Amazon EC2 : autorise le démarrage ou l'arrêt des instances EC2 balisées par un utilisateur, par programmation et dans la console
- EC2 : démarrer ou arrêter les instances en fonction des balises de ressources et de principaux correspondantes
- EC2: démarrer ou arrêter des instances en fonction des balises
- IAM : endosser des rôles qui ont une balise spécifique

Politique ABAC : accéder aux ressources Secrets Manager uniquement lorsque les balises du principal et des ressources correspondent

La politique suivante autorise les principaux à créer, lire, modifier et supprimer des ressources, mais uniquement lorsque ces ressources sont balisées avec les mêmes paires clé-valeur que le principal. Lorsqu'un principal crée une ressource, il doit ajouter les balises `access-project`, `access-team` et `cost-center` avec des valeurs qui correspondent aux balises du principal. La politique autorise également l'ajout de balises `Name` ou `OwnedBy` facultatives.

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
     {
         "Sid": "AllActionsSecretsManagerSameProjectSameTeam",
         "Effect": "Allow",
         "Action": "secretsmanager:*",
         "Resource": "*",
         "Condition": {
             "StringEquals": {
                 "aws:ResourceTag/access-project": "${aws:PrincipalTag/access-
project}",
                 "aws:ResourceTag/access-team": "${aws:PrincipalTag/access-
team}",
                 "aws:ResourceTag/cost-center": "${aws:PrincipalTag/cost-center}"
             },
             "ForAllValues:StringEquals": {
                 "aws:TagKeys": [
                     "access-project",
                     "access-team",
```

```
                    "cost-center",
                    "Name",
                    "OwnedBy"
                ]
            },
            "StringEqualsIfExists": {
                "aws:RequestTag/access-project": "${aws:PrincipalTag/access-
project}",
                "aws:RequestTag/access-team": "${aws:PrincipalTag/access-team}",
                "aws:RequestTag/cost-center": "${aws:PrincipalTag/cost-center}"
            }
        }
    },
    {
        "Sid": "AllResourcesSecretsManagerNoTags",
        "Effect": "Allow",
        "Action": [
            "secretsmanager:GetRandomPassword",
            "secretsmanager:ListSecrets"
        ],
        "Resource": "*"
    },
    {
        "Sid": "ReadSecretsManagerSameTeam",
        "Effect": "Allow",
        "Action": [
            "secretsmanager:Describe*",
            "secretsmanager:Get*",
            "secretsmanager:List*"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/access-team": "${aws:PrincipalTag/access-team}"
            }
        }
    },
    {
        "Sid": "DenyUntagSecretsManagerReservedTags",
        "Effect": "Deny",
        "Action": "secretsmanager:UntagResource",
        "Resource": "*",
        "Condition": {
            "ForAnyValue:StringLike": {
```

```
                "aws:TagKeys": "access-*"
            }
        }
    },
    {
        "Sid": "DenyPermissionsManagement",
        "Effect": "Deny",
        "Action": "secretsmanager:*Policy",
        "Resource": "*"
    }
 ]
}
```

À quoi sert cette politique ?

- La instruction `AllActionsSecretsManagerSameProjectSameTeam` autorise toutes les actions de ce service sur toutes les ressources associées, mais uniquement si les balises des ressources correspondent aux balises du principal. En ajoutant `"Action": "secretsmanager:*"` à la politique, cette dernière évolue en même temps que Secrets Manager. Si Secrets Manager ajoute une nouvelle opération d'API, vous n'êtes pas obligé d'ajouter cette action à l'instruction. La instruction implémente l'ABAC en utilisant trois blocs de condition. La demande n'est autorisée que si les trois blocs renvoient la valeur « true » (vrai).
  - Le premier bloc de condition de cette instruction renvoie la valeur true si les clés de balise spécifiées sont présentes sur la ressource et que leurs valeurs correspondent aux balises du principal. Ce bloc renvoie la valeur « false » (faux) pour les étiquettes qui ne correspondent pas, ou pour les actions qui ne prennent pas en charge l'étiquettage des ressources. Pour savoir quelles actions ne sont pas autorisées par ce bloc, consultez la section Actions, ressources et clés de condition pour AWS Secrets Manager. Cette page montre que les actions effectuées sur le type de ressource Secret prennent en charge la clé de condition `secretsmanager:ResourceTag/tag-key`. Certaines actions Secrets Manager ne prennent pas en charge ce type de ressource, y compris `GetRandomPassword` et `ListSecrets`. Vous devez créer des instructions supplémentaires pour autoriser ces actions.
  - Le deuxième bloc de condition renvoie la valeur « true » (vrai) si chaque clé d'identification transmise dans la demande figure dans la liste spécifiée. Cette opération est effectuée en utilisant `ForAllValues` avec l'opérateur de condition `StringEquals`. Si aucune clé ni sous-ensemble de l'ensemble de clés n'est transmis, la condition renvoie la valeur « true » (vrai). Cela permet les opérations `Get*` qui n'autorisent pas l'inclusion des étiquettes dans la demande. Si le demandeur inclut une clé d'identification qui ne figure pas dans la liste, la condition renvoie la

valeur « false » (faux) Chaque clé de balise transmise dans la demande doit correspondre à un membre de cette liste. Pour de plus amples informations, veuillez consulter Définir les opérateurs pour les clés de contexte à valeurs multiples.

- Le troisième bloc de condition renvoie la valeur « true » (vrai) si la demande prend en charge la transmission des étiquettes, si les trois étiquettes sont présentes et si elles correspondent aux valeurs de l'étiquette principale. Ce bloc renvoie également la valeur « true » (vrai) si la demande ne prend pas en charge la transmission des étiquettes. C'est grâce à `...IfExists` dans l'opérateur de condition. Le bloc renvoie la valeur « false » (faux) si aucune étiquette n'est transmise pendant une action qui la prend en charge, ou si les clés et les valeurs d'étiquette ne correspondent pas.

- La instruction `AllResourcesSecretsManagerNoTags` autorise les actions `GetRandomPassword` et `ListSecrets` qui ne sont pas autorisées par la première instruction.
- La instruction `ReadSecretsManagerSameTeam` autorise les opérations en lecture seule si le principal est balisé avec la même balise access-team que la ressource. Ceci est autorisé indépendamment de l'étiquette du projet ou du centre de coûts.
- La instruction `DenyUntagSecretsManagerReservedTags` refuse les demandes de suppression de balises avec des clés commençant par « access- » de Secrets Manager. Ces balises servent à contrôler l'accès aux ressources. Leur suppression pourrait supprimer des autorisations.
- La déclaration `DenyPermissionsManagement` refuse l'accès à la création, à la modification ou à la suppression de politiques basées sur les ressources de Secrets Manager. Ces politiques peuvent être utilisées pour modifier les autorisations du secret.

> **Important**
>
> Cette politique utilise une stratégie qui autorise toutes les actions d'un service, mais refuse explicitement les actions de modification des autorisations. Le refus d'une action remplace toute autre politique autorisant le principal à effectuer cette action. Ce refus peut entraîner des résultats imprévus. La bonne pratique consiste à utiliser uniquement le refus explicite lorsqu'aucune circonstance ne doit autoriser cette action. Sinon, dressez la liste des actions autorisées. Les actions indésirables seront refusées par défaut.

Étape 3 : Créer les rôles

Créez les rôles IAM suivants et attachez la politique **`access-same-project-team`** que vous avez créée à l'étape précédente. Pour plus d'informations sur la création de rôles IAM, veuillez consulter Créer un rôle pour attribuer des autorisations à un utilisateur IAM. Si vous sélectionnez d'utiliser la fédération plutôt que des utilisateurs et des rôles IAM, veuillez consulter Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC.

Fonction de tâche	Nom du rôle	Balises de rôle	Description du rôle
Ingénierie du projet Pegasus	access-peg-engineering	access-project = `peg` access-team = `eng` cost-center = `987654`	Autorise les ingénieurs à lire toutes les ressources d'ingénierie et à créer et gérer les ressources d'ingénierie du projet Pegasus.
Assurance qualité du projet Pegasus	access-peg-quality-assurance	access-project = `peg` access-team = `qas` cost-center = `987654`	Autorise l'équipe d'assurance qualité à lire toutes les ressources d'assurance qualité et à créer et gérer toutes les ressources d'assurance qualité du projet Pegasus.
Ingénierie du projet Unicorn	access-uni-engineering	access-project = `uni` access-team = `eng`	Autorise les ingénieurs à lire toutes les

Fonction de tâche	Nom du rôle	Balises de rôle	Description du rôle
		cost-center = 123456	ressources d'ingénierie et à créer et gérer les ressources d'ingénierie du projet Unicorn.
Assurance qualité du projet Unicorn	access-uni-quality-assurance	access-project = uni access-team = qas cost-center = 123456	Autorise l'équipe d'assurance qualité à lire toutes les ressources d'assurance qualité et à créer et gérer toutes les ressources d'assurance qualité du projet Unicorn.

Étape 4 : Tester la création de secrets

La politique d'autorisation attachée aux rôles autorise les employés à créer des secrets. Ceci n'est autorisé que si le secret est labelisé au niveau du projet, de l’équipe et du centre de coûts. Vérifiez que vos autorisations fonctionnent comme prévu en vous connectant comme vos utilisateurs, en endossant le bon rôle et en testant l'activité dans Secrets Manager.

Pour tester la création d'un secret avec et sans les balises requises

1. Dans la fenêtre principale de votre navigateur, restez connecté en tant qu'utilisateur administrateur afin que vous puissiez passer en revue les utilisateurs, les rôles et les politiques dans IAM. Utilisez une fenêtre de navigation privée ou un navigateur séparé pour votre test. Connectez-vous en tant qu'utilisateur access-Arnav-peg-eng IAM et ouvrez la console Secrets Manager à https://console.aws.amazon.com/secretsmanager/ l'adresse.

2. Tentez de basculer vers le rôle `access-uni-engineering`.

   Cette opération échoue, car les valeurs de balise `access-project` et `cost-center` ne correspondent pas à l'utilisateur `access-Arnav-peg-eng` et au rôle `access-uni-engineering`.

   Pour plus d'informations sur le changement de rôle dans le Console de gestion AWS, voir Basculer d'un utilisateur à un rôle IAM (console)

3. Basculez vers le rôle `access-peg-engineering`.

4. Enregistrez un nouveau secret en utilisant les informations suivantes. Pour savoir comment enregistrer un secret, veuillez consulter Création d'un secret basique dans le Guide de l'utilisateur AWS Secrets Manager .

   > **Important**
   >
   > Secrets Manager affiche des alertes indiquant que vous ne disposez pas d'autorisations pour des services AWS supplémentaires qui fonctionnent avec Secrets Manager. Par exemple, pour créer des informations d'identification pour une base de données Amazon RDS, vous devez être autorisé à décrire des instances RDS, des clusters RDS et des clusters Amazon Redshift. Vous pouvez ignorer ces alertes puisque vous n'utilisez pas ces AWS services spécifiques dans ce didacticiel.

   1. Dans la section Sélectionner un type de secret sélectionnez Autre type de secrets. Dans les deux zones de texte, saisissez `test-access-key` et `test-access-secret`.
   2. Saisissez `test-access-peg-eng` dans le champ Nom du secret .
   3. Ajoutez différentes combinaisons de balises du tableau suivant et visualisez le comportement attendu.
   4. Choisissez Stocker pour tenter de créer le secret. Si le stockage échoue, revenez aux pages précédentes de la console Secrets Manager et utilisez l'ensemble de balises suivant dans le tableau ci-dessous. Le dernier ensemble de balises est autorisé et crée le secret avec succès.

   Le tableau suivant présente les combinaisons de balises ABAC pour le rôle `test-access-peg-eng`.

access-project Valeur de l'étiquette	**access-team** Valeur de l'étiquette	**cost-center** Valeur de l'étiquette	Balises supplémentaires	Comportement attendu
(aucun)	(aucun)	(aucun)	(aucun)	Refusé, car la valeur de la balise `access-project` ne correspond pas à la valeur `peg` du rôle.
`uni`	`eng`	`987654`	(aucun)	Refusé, car la valeur de la balise `access-project` ne correspond pas à la valeur `peg` du rôle.
`peg`	`qas`	`987654`	(aucun)	Refusé, car la valeur de la balise `access-team` ne correspond pas à la valeur `eng` du rôle.
`peg`	`eng`	`123456`	(aucun)	Refusé, car la valeur de la balise `cost-center` ne correspond pas à la valeur `987654` du rôle.
`peg`	`eng`	`987654`	Propriétaire = Jane	Refusé, car la balise `owner` supplémentaire n'est pas autorisée par la politique, même si les trois balises requises sont présentes et que leurs valeurs correspondent à celles du rôle.
`peg`	`eng`	`987654`	Nom = Jane	Autorisé, car les trois balises obligatoires sont présentes et leurs valeurs correspondent aux valeurs du rôle. Vous êtes également autorisé à inclure la balise `Name` facultative.

5. Déconnectez-vous et répétez les trois premières étapes de cette procédure pour chaque rôle et chaque valeur de balise ci-dessous. À la quatrième étape de cette procédure, testez tous

les ensembles de balises manquantes, de balises facultatives, de balises non autorisées et de valeurs de balises non valides de votre choix. Puis, utilisez les balises requises pour créer un secret avec les balises et le nom suivants.

Nom utilisateur	Nom du rôle	Nom du secret	Balises du secret
accès- Mary-peg-qas	access-peg-quality-assurance	test-access-peg-qas	access-project = `peg` access-team = `qas` cost-center = `987654`
accès- Saanvi-uni-eng	access-uni-engineering	test-access-uni-eng	access-project = `uni` access-team = `eng` cost-center = `123456`
accès- Carlos-uni-qas	access-uni-quality-assurance	test-access-uni-qas	access-project = `uni` access-team = `qas` cost-center = `123456`

Étape 5 : Tester l'affichage des secrets

La politique que vous avez attachée à chaque rôle autorise les employés à afficher tous les secrets labelisés avec leur nom d'équipe, quel que soit leur projet. Vérifiez que vos autorisations fonctionnent comme prévu en testant vos rôles dans Secrets Manager.

Pour tester l'affichage d'un secret avec et sans les balises requises

1. Connectez-vous en tant que l'un des utilisateurs IAM suivants :
   - `access-Arnav-peg-eng`
   - `access-Mary-peg-qas`

- `access-Saanvi-uni-eng`
- `access-Carlos-uni-qas`

2. Basculez vers le rôle correspondant :

   - `access-peg-engineering`
   - `access-peg-quality-assurance`
   - `access-uni-engineering`
   - `access-uni-quality-assurance`

   Pour plus d'informations sur le changement de rôle dans le Console de gestion AWS, consultezBasculer d'un utilisateur à un rôle IAM (console).

3. Dans le panneau de navigation de gauche, sélectionnez l'icône de menu permettant de développer le menu, puis sélectionnez Secrets.
4. Vous devriez voir les quatre secrets du tableau, quel que soit votre rôle actuel. C'est le comportement attendu, car la politique nommée `access-same-project-team` autorise l'action `secretsmanager:ListSecrets` pour toutes les ressources.
5. Choisissez l'un des secrets.
6. Sur la page détaillée du secret, les étiquettes de votre rôle déterminent si vous pouvez afficher le contenu de la page. Comparez le nom de votre rôle avec celui de votre secret. S'ils ont le même nom d'équipe, alors les balises `access-team` correspondent. Si elles ne correspondent pas, l'accès est refusé.

   Le tableau suivant montre le comportement d'affichage du secret ABAC pour chaque rôle.

Nom du rôle	Nom du secret	Comportement attendu
`access-peg-engineering`	`test-access-peg-eng`	Autorisé
	test-access-peg-qas	Refusé
	test-access-uni-eng	Autorisé
	test-access-uni-qas	Refusé
access-peg-quality-assurance	test-access-peg-eng	Refusé

Nom du rôle	Nom du secret	Comportement attendu
	test-access-peg-qas	Autorisé
	test-access-uni-eng	Refusé
	test-access-uni-qas	Autorisé
access-uni-engineering	test-access-peg-eng	Autorisé
	test-access-peg-qas	Refusé
	test-access-uni-eng	Autorisé
	test-access-uni-qas	Refusé
access-uni-quality-assurance	test-access-peg-eng	Refusé
	test-access-peg-qas	Autorisé
	test-access-uni-eng	Refusé
	test-access-uni-qas	Autorisé

7. Dans les miniatures en haut de la page, sélectionnez Secrets pour revenir à la liste des secrets. Répétez les étapes de cette procédure en utilisant différents rôles pour vérifier si vous pouvez afficher chacun des secrets.

Étape 6 : Tester l'adaptabilité

L'évolutivité est une raison majeure de privilégier le contrôle d'accès basé sur les attributs (ABAC) par rapport au contrôle d'accès basé sur les rôles (RBAC). Au fur et à mesure que votre entreprise ajoute de nouveaux projets, équipes ou personnes AWS, vous n'avez pas besoin de mettre à jour vos ABAC-driven politiques. Par exemple, supposons que la société Exemple finance un nouveau projet, dont le nom de code est Centaur. Un ingénieur nommé Saanvi Sarkar sera l'ingénieur principal du projet Centaur tout en continuant à travailler sur le projet Unicorn . Saanvi passera également en revue les travaux du projet Pegasus. Plusieurs ingénieurs embauchés récemment, dont Nikhil Jayashankar, travailleront exclusivement sur le projet Centaur .

Pour ajouter le nouveau projet à AWS

1. Connectez-vous en tant qu'administrateur IAM et ouvrez la console IAM à l'adresse. https://console.aws.amazon.com/iam/
2. Dans le panneau de navigation de gauche, sélectionnez Roles (Rôles) et ajoutez un rôle IAM nommé `access-cen-engineering`. Attachez la politique d'autorisation **`access-same-project-team`** au rôle et ajoutez les balises de rôle suivantes :
   - `access-project` = `cen`
   - `access-team` = `eng`
   - `cost-center` = `101010`
3. Dans le volet de navigation de gauche, choisissez Utilisateurs.
4. Ajoutez un nouvel utilisateur nommé `access-Nikhil-cen-eng`, attachez la politique nommée `access-assume-role`, et ajoutez les balises utilisateur suivantes.
   - `access-project` = `cen`
   - `access-team` = `eng`
   - `cost-center` = `101010`
5. Utilisez les procédures des sections Étape 4 : Tester la création de secrets et Étape 5 : Tester l'affichage des secrets. Dans une autre fenêtre de navigateur, vérifiez que Nikhil ne peut créer que des secrets d'ingénierie pour le projet Centaur et qu'il peut afficher tous les secrets d'ingénierie.
6. Dans la fenêtre principale du navigateur à partir de laquelle vous vous êtes connecté en tant qu'administrateur, sélectionnez l'utilisateur `access-Saanvi-uni-eng`.
7. Sous l'onglet Autorisations, supprimez la politique d'autorisation access-endosse-role .
8. Ajoutez la politique en ligne ci-dessous et nommez-la `access-assume-specific-roles`. Pour de plus amples informations sur l'intégration d'une politique en ligne pour un utilisateur, veuillez consulter Pour intégrer une politique en ligne pour un utilisateur ou un rôle (console).

   Politique ABAC : endosser uniquement des rôles spécifiques

   Cette politique autorise Saanvi à assumer les rôles d'ingénierie pour les projets Pegasus et Centaure. Il est nécessaire de créer cette politique personnalisée, car IAM ne prend pas en charge les balises à valeurs multiples. Vous ne pouvez pas baliser l'utilisateur Saanvi avec `access-project` = `peg` et `access-project` = `cen`. De plus, le modèle AWS d'autorisation ne peut pas correspondre aux deux valeurs. Pour de plus amples informations, veuillez consulter

Règles de balisage dans IAM et AWS STS. Vous devez spécifier manuellement les deux rôles qu'elle peut endosser.

Pour utiliser cette politique, remplacez le texte en italique de l'espace réservé dans l'exemple de politique par vos propres informations de compte.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TutorialAssumeSpecificRoles",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": [
                "arn:aws:iam::111122223333:role/access-peg-engineering",
                "arn:aws:iam::111122223333:role/access-cen-engineering"
            ]
        }
    ]
}
```

9. Utilisez les procédures des sections Étape 4 : Tester la création de secrets et Étape 5 : Tester l'affichage des secrets. Dans une autre fenêtre du navigateur, vérifiez que Saanvi peut endosser les deux rôles. Vérifiez qu'elle peut créer des secrets uniquement pour son projet, son équipe et son centre de coûts, en fonction des balises du rôle. Vérifiez également qu'elle peut afficher les détails de tous les secrets appartenant à l'équipe d'ingénierie, y compris ceux qu'elle vient de créer.

Étape 7 : Tester la mise à jour et la suppression des secrets

La politique `access-same-project-team` attachée aux rôles autorise les employés à mettre à jour et à supprimer tous les secrets possédant les mêmes balises que leur projet, leur équipe et leur centre de coûts. Vérifiez que vos autorisations fonctionnent comme prévu en testant vos rôles dans Secrets Manager.

Pour tester la mise à jour et la suppression d'un secret avec et sans les balises requises

1. Connectez-vous en tant que l'un des utilisateurs IAM suivants :

- `access-Arnav-peg-eng`
- `access-Mary-peg-qas`
- `access-Saanvi-uni-eng`
- `access-Carlos-uni-qas`
- `access-Nikhil-cen-eng`

2. Basculez vers le rôle correspondant :

   - `access-peg-engineering`
   - `access-peg-quality-assurance`
   - `access-uni-engineering`
   - `access-peg-quality-assurance`
   - `access-cen-engineering`

   Pour plus d'informations sur le changement de rôle dans le Console de gestion AWS, consultezBasculer d'un utilisateur à un rôle IAM (console).

3. Pour chaque rôle, mettez à jour la description du secret, puis supprimez les secrets suivants. Pour de plus amples informations, veuillez consulter Modification d'un secret et Suppression et restauration d'un secret dans le Guide de l'utilisateur AWS Secrets Manager .

   Le tableau suivant montre le comportement de mise à jour et de suppression du secret ABAC pour chaque rôle.

Nom du rôle	Nom du secret	Comportement attendu
access-peg-engineering	test-access-peg-eng	Autorisé
	test-access-uni-eng	Refusé
	test-access-uni-qas	Refusé
access-peg-quality-assuranc e	test-access-peg-qas	Autorisé
	test-access-uni-eng	Refusé
access-uni-engineering	test-access-uni-eng	Autorisé

Nom du rôle	Nom du secret	Comportement attendu
	test-access-uni-qas	Refusé
access-peg-quality-assurance	test-access-uni-qas	Autorisé

Résumé

Vous avez à présent terminé toutes les étapes nécessaires pour utiliser des balises pour le contrôle d'accès basé sur les attributs (ABAC). Vous avez appris à définir une stratégie de balisage. Vous avez appliqué cette stratégie à vos principaux et à vos ressources. Vous avez créé et appliqué une politique qui met exécute la stratégie pour Secrets Manager. Vous avez également appris que l'ABAC évolue facilement lorsque vous ajoutez de nouveaux projets et membres d'équipe. Désormais, vous pouvez vous connecter à la console IAM avec vos rôles test et tester l'utilisation de balises ABAC dans AWS.

> **Note**
>
> Vous avez ajouté des politiques qui autorisent les actions uniquement dans des conditions spécifiques. Si vous appliquez une politique différente à vos utilisateurs ou rôles disposant d'autorisations plus larges, il se peut que les actions ne soient pas limitées pour demander le balisage. Par exemple, si vous accordez à un utilisateur des autorisations administratives complètes `AdministratorAccess` AWS à l'aide de la politique gérée, ces politiques ne limitent pas cet accès. Pour plus d'informations sur la façon dont les autorisations sont déterminées lorsque plusieurs politiques sont impliquées, veuillez consulter Comment ? AWS la logique du code d'application évalue les demandes pour autoriser ou refuser l'accès.

Ressources connexes

Pour plus d'informations, consultez les ressources suivantes :

- Définition des autorisations en fonction des attributs avec autorisation ABAC
- AWS clés de contexte de condition globale
- Créer un rôle pour attribuer des autorisations à un utilisateur IAM
- Tags pour Gestion des identités et des accès AWS resources

- Contrôle de l'accès aux AWS ressources à l'aide de balises
- Basculer d'un utilisateur à un rôle IAM (console)
- Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC

Pour savoir comment surveiller les balises de votre compte, consultez Surveiller les modifications de balises sur les AWS ressources à l'aide de flux de travail sans serveur et Amazon CloudWatch Events.

Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC

Attribute-based le contrôle d'accès (ABAC) est une stratégie d'autorisation qui définit les autorisations en fonction d'attributs. Dans AWS, ces attributs sont appelés balises. Vous pouvez associer des balises aux ressources IAM, y compris aux entités IAM (utilisateurs ou rôles), et aux AWS ressources. Lorsque les entités sont utilisées pour adresser des requêtes AWS, elles deviennent des mandants et ces derniers incluent des balises.

Vous pouvez également transmettre des balises de session lorsque vous endossez un rôle ou fédérez un utilisateur. Vous pouvez ensuite définir des politiques qui utilisent des clés de condition de balise pour accorder des autorisations à vos principaux en fonction de leurs balises. Lorsque vous utilisez des balises pour contrôler l'accès à vos ressources AWS , vous autorisez vos équipes et vos ressources à se développer en modifiant moins les politiques AWS . Les politiques ABAC sont plus flexibles que AWS les politiques traditionnelles, qui vous obligent à répertorier chaque ressource individuelle. Pour de plus amples informations sur l'ABAC et ses avantages par rapport aux politiques traditionnelles, veuillez consulter Définition des autorisations en fonction des attributs avec autorisation ABAC.

Si votre entreprise utilise un fournisseur d' SAML-based identité (IdP) pour gérer les identités des utilisateurs d'entreprise, vous pouvez utiliser les attributs SAML pour un contrôle d'accès précis dans. AWS Les attributs peuvent inclure des identifiants de centre de coûts, des adresses e-mail d'utilisateurs, des classifications de services et des affectations de projet. Lorsque vous transmettez ces attributs en tant qu'étiquettes de session, vous pouvez ensuite contrôler l'accès à l'interface AWS en fonction de ces étiquettes de session.

Pour réaliser le didacticiel de l'ABAC en transmettant les attributs SAML à votre principal de session, effectuez les tâches de la section Tutoriel IAM : définir les autorisations d'accès AWS ressources basées sur des balises avec les modifications citées dans cette rubrique.

Conditions préalables

Pour réaliser les étapes d'utilisation des étiquettes de session SAML pour l'ABAC, vous devez déjà disposer des éléments suivants :

- Accès à un SAML-based IdP où vous pouvez créer des utilisateurs de test dotés d'attributs spécifiques.
- La possibilité de se connecter en tant qu'utilisateur disposant d'autorisations d'administrateur.
- Une expérience dans la création et la modification d'utilisateurs, de rôles et de politiques IAM dans Console de gestion AWS. Toutefois, si vous avez besoin d'aide pour mémoriser un processus de gestion IAM, le didacticiel de l'ABAC fournit des liens vers les instructions étape par étape.
- Expérience de la configuration d'un SAML-based IdP dans IAM. Pour afficher de plus amples informations et des liens vers la documentation détaillée IAM, veuillez consulter Transmettre des balises de session à l'aide AssumeRoleWithSAML.

Étape 1 : Créer des utilisateurs test

Ignorez les instructions de la section Étape 1 : Créer des utilisateurs test. Étant donné que vos identités sont définies dans votre fournisseur, il n'est pas nécessaire que vous ajoutiez des utilisateurs IAM pour vos employés.

Étape 2 : Créer la politique ABAC

Suivez les instructions de la section Étape 2 : Créer la politique ABAC pour créer la politique gérée spécifiée dans IAM.

Étape 3 : Créer et configurer le rôle SAML

Lorsque vous utilisez le didacticiel ABAC pour SAML, vous devez effectuer des étapes supplémentaires pour créer le rôle, configurer l'IdP SAML et activer l'accès. Console de gestion AWS Pour de plus amples informations, veuillez consulter Étape 3 : Créer les rôles.

Étape 3A : Créer le rôle SAML

Créez un rôle unique qui approuve votre fournisseur d'identité SAML et l'utilisateur `test-session-tags` que vous avez créé à l'étape 1. Le didacticiel de l'ABAC utilise des rôles distincts avec des balises de rôle différentes. Étant donné que vous transmettez des balises de session à partir de votre

fournisseur d'identité SAML, vous n'avez besoin que d'un seul rôle. Pour savoir comment créer un SAML-based rôle, consultezCréation d'un rôle pour la fédération SAML 2.0 (console).

Nommez le rôle `access-session-tags`. Attachez la politique d'autorisation `access-same-project-team` au rôle. Modifiez la politique d'approbation de rôle pour utiliser la politique suivante. Pour obtenir des instructions détaillées sur la modification de la relation d'approbation d'un rôle, veuillez consulter Mise à jour d'une politique d'approbation de rôle.

La politique d'approbation de rôle suivante permet à votre fournisseur d'identité SAML et à l'utilisateur `test-session-tags` d'endosser le rôle. Lorsqu'ils endossent le rôle, ils doivent transmettre les trois balises de session spécifiées. L'action `sts:TagSession` est requise pour autoriser la transmission des balises de session.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSamlIdentityAssumeRole",
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRoleWithSAML",
                "sts:TagSession"
            ],
            "Principal": {"Federated":"arn:aws:iam::123456789012:saml-
provider/ExampleCorpProvider"},
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/cost-center": "*",
                    "aws:RequestTag/access-project": "*",
                    "aws:RequestTag/access-team": [
                        "eng",
                        "qas"
                    ]
                },
                "StringEquals": {"SAML:aud": "https://region-
code.signin.aws.amazon.com/saml"}
            }
        }
    ]
```

```
}
```

La `AllowSamlIdentityAssumeRole` déclaration permet aux membres des équipes d'ingénierie et d'assurance qualité d'assumer ce rôle lorsqu'ils se fédérent à AWS partir de l'IdP d'Example Corporation. Le fournisseur SAML `ExampleCorpProvider` est défini dans IAM. L'administrateur a déjà configuré l'assertion SAML pour transmettre les trois balises de session requises. L'assertion peut transmettre des balises supplémentaires, mais ces trois balises doivent être présentes. Les attributs de l'identité peuvent avoir n'importe quelle valeur pour les balises `cost-center` et `access-project`. Toutefois, la valeur de l'attribut `access-team` doit correspondre à `eng` ou `qas` pour indiquer que l'identité se trouve dans l'équipe d'ingénierie ou d'assurance qualité.

Étape 3B : Configurer le fournisseur d'identité SAML

Configurez votre fournisseur d'identité SAML pour qu'il transmette les attributs `cost-center`, `access-project` et `access-team` en tant que balises de session. Pour de plus amples informations, veuillez consulter Transmettre des balises de session à l'aide AssumeRoleWithSAML.

Pour transmettre ces attributs en tant que balises de session, incluez les éléments suivants dans votre assertion SAML.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:cost-center">
  <AttributeValue>987654</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:access-project">
  <AttributeValue>peg</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:access-team">
  <AttributeValue>eng</AttributeValue>
</Attribute>
```

Étape 3C : Activer l'accès à la console

Activez l'accès à la console pour vos utilisateurs SAML fédérés. Pour de plus amples informations, veuillez consulter Permettre aux principaux fédérés SAML 2.0 d'accéder au Console de gestion AWS.

Étape 4 : Tester la création de secrets

Fédérez-vous pour Console de gestion AWS utiliser le `access-session-tags` rôle. Pour de plus amples informations, veuillez consulter Permettre aux principaux fédérés SAML 2.0 d'accéder au

Console de gestion AWS. Puis, suivez les instructions de la section Étape 4 : Tester la création de secrets pour créer des secrets. Utilisez différentes identités SAML avec des attributs pour correspondre aux balises indiquées dans le didacticiel de l'ABAC. Pour plus d'informations, veuillez consulter Étape 4 : Tester la création de secrets.

Étape 5 : Tester l'affichage des secrets

Suivez les instructions de la section Étape 5 : Tester l'affichage des secrets pour afficher les secrets que vous avez créés à l'étape précédente. Utilisez différentes identités SAML avec des attributs pour correspondre aux balises indiquées dans le didacticiel de l'ABAC.

Étape 6 : Tester l'adaptabilité

Suivez les instructions de la section Étape 6 : Tester l'adaptabilité pour tester l'évolutivité. Pour ce faire, ajoutez une nouvelle identité à votre SAML-based IdP avec les attributs suivants :

- `cost-center = 101010`
- `access-project = cen`
- `access-team = eng`

Étape 7 : Tester la mise à jour et la suppression des secrets

Suivez les instructions de l'étape Étape 7 : Tester la mise à jour et la suppression des secrets pour mettre à jour et supprimer des secrets. Utilisez différentes identités SAML avec des attributs pour correspondre aux balises indiquées dans le didacticiel de l'ABAC.

> **Important**
>
> Supprimez tous les secrets que vous avez créés pour éviter les frais de facturation. Pour de plus amples informations sur la tarification de Secrets Manager, veuillez consulter Tarification AWS Secrets Manager.

Résumé

Vous avez maintenant terminé avec succès toutes les étapes nécessaires pour utiliser les balises de session SAML et les balises de ressources pour la gestion des autorisations.

> **Note**
>
> Vous avez ajouté des politiques qui autorisent les actions uniquement dans des conditions spécifiques. Si vous appliquez une politique différente à vos utilisateurs ou rôles disposant d'autorisations plus larges, il se peut que les actions ne soient pas limitées pour demander le balisage. Par exemple, si vous accordez à un utilisateur des autorisations administratives complètes `AdministratorAccess` AWS à l'aide de la politique gérée, ces politiques ne limitent pas cet accès. Pour plus d'informations sur la façon dont les autorisations sont déterminées lorsque plusieurs politiques sont impliquées, veuillez consulter Comment ? AWS la logique du code d'application évalue les demandes pour autoriser ou refuser l'accès.

Didacticiel IAM : permettre aux utilisateurs de gérer leurs informations d'identification et leurs paramètres MFA

Vous pouvez autoriser les utilisateurs à gérer eux-mêmes leurs propres informations d'identification et dispositifs d'authentification multifactorielle (MFA) sur la page Informations d'identification de sécurité. Vous pouvez les utiliser Console de gestion AWS pour configurer les informations d'identification (clés d'accès, mots de passe, certificats de signature et clés publiques SSH), supprimer ou désactiver les informations d'identification inutiles et activer les dispositifs MFA pour vos utilisateurs. Cette méthode est utile pour un petit nombre d'utilisateurs, mais la tâche peut rapidement devenir fastidieuse si le nombre d'utilisateurs augmente. Ce didacticiel vous montre comment mettre en place ces bonnes pratiques sans surcharger de travail vos administrateurs.

Ce didacticiel explique comment autoriser les utilisateurs à accéder aux AWS services, mais uniquement lorsqu'ils se connectent à l'aide de la MFA. S'ils ne sont pas connectés avec un dispositif MFA, les utilisateurs ne peuvent pas accéder à d'autres services.

Ce flux de travail se compose de trois étapes de base.

Étape 1 : Créer une politique pour appliquer l'authentification MFA

Créez une politique gérée par le client qui interdit toutes les actions à l'exception des quelques actions IAM. Ces exceptions permettent à un utilisateur de modifier ses propres informations d'identification et de gérer ses dispositifs MFA sur la page Informations d'identification de sécurité. Plus d'informations sur la manière d'accéder à cette page, veuillez consulter Comment les utilisateurs IAM modifient leur mot de passe (console).

Étape 2 : Attacher des politiques à votre groupe d'utilisateurs test

Créez un groupe d'utilisateurs dont les membres ont un accès total à toutes les actions Amazon EC2 s'ils se connectent avec MFA. Pour créer un tel groupe d'utilisateurs, vous devez associer à la fois la politique AWS gérée appelée `AmazonEC2FullAccess` et la politique gérée par le client que vous avez créée lors de la première étape.

Étape 3 : Test de l'accès utilisateur

Connectez-vous en tant qu'utilisateur test pour vérifier que l'accès à Amazon EC2 est bloqué jusqu'à ce que l'utilisateur crée un dispositif MFA. L'utilisateur peut alors se connecter à l'aide de ce dispositif.

Conditions préalables

Pour exécuter les étapes de ce didacticiel, vous devez déjà disposer des éléments suivants :

- Et Compte AWS auquel vous pouvez vous connecter en tant qu'utilisateur IAM avec des autorisations administratives.
- Votre ID de compte, que vous entrez dans la politique à l'étape 1.

  Pour trouver votre ID de compte, sur la barre de navigation située en haut de la page, sélectionnez Support, puis sélectionnez Support Center (Centre de support). L'ID de compte se trouve dans le menu Support de cette page.
- Un périphérique MFA virtuel (logiciel), une clé de sécurité FIDO ou un dispositif MFA matériel.
- Un utilisateur IAM test et membre du groupe d'utilisateurs suivant :

Nom utilisateur	Instructions du nom d'utilisateur	Nom du groupe d'utilisateurs	Ajouter l'utilisateur en tant que membre	Instructions pour les groupes d'utilisateurs
MFAUser	Choisissez uniquement l'option pour Activer l'accès à la console (facultatif) et attribuez un mot de passe.	EC2MFA	MFAUser	N'attachez PAS de politique ni n'accordez d'autorisations à ce groupe d'utilisateurs.

Étape 1 : Créer une politique pour appliquer l'authentification MFA

Vous commencez par créer une politique gérée par le client IAM qui refuse toutes les autorisations, sauf celles requises par les utilisateurs IAM pour gérer leurs informations d'identification et appareils MFA.

1. Connectez-vous à la console de AWS gestion en tant qu'utilisateur avec des informations d'identification d'administrateur. Pour respecter les meilleures pratiques IAM, ne vous connectez pas avec vos Utilisateur racine d'un compte AWS informations d'identification.

   > **Important**
   >
   > Les meilleures pratiques IAM recommandent de demander aux utilisateurs humains d'utiliser la fédération avec un fournisseur d'identité pour accéder à l'aide d'informations d'identification temporaires plutôt que d' AWS utiliser des utilisateurs IAM dotés d'informations d'identification à long terme. Nous vous recommandons de n'utiliser les utilisateurs IAM que pour les cas d'utilisation spécifiques non pris en charge par les utilisateurs fédérés.

2. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.

3. Dans le panneau de navigation, sélectionnez Policies (Politiques), puis Create policy (Créer une politique).

4. Sélectionnez l'onglet JSON, et copiez le texte du document de politique JSON suivant : AWS : autorise les utilisateurs IAM authentifiés par MFA à gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

5. Collez cette politique dans la zone de texte JSON. Résolvez tous les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant.

   > **Note**
   >
   > Vous pouvez basculer à tout moment entre les options Éditeur visuel et JSON. Cependant, la politique ci-dessus inclut l'élément `NotAction` qui n'est pas pris en charge dans l'éditeur visuel. Pour cette politique, vous verrez une notification dans l'onglet Visual editor (Éditeur visuel). Revenez à JSON pour continuer à travailler avec cette politique.

> Cet exemple de politique n'autorise pas les utilisateurs à réinitialiser un mot de passe lorsqu'ils se connectent à l' Console de gestion AWS pour la première fois. Nous vous recommandons de n'accorder des autorisations aux nouveaux utilisateurs qu'après leur enregistrement et aient réinitialisé leur mot de passe.

6. Sur la page Vérifier et créer, tapez **`Force_MFA`** pour le nom de la politique. Pour la description de la politique, tapez **`This policy allows users to manage their own passwords and MFA devices but nothing else unless they authenticate with MFA.`** dans la zone Balises ; vous pouvez éventuellement ajouter des paires clé-valeur de balises à la politique gérée par le client. Vérifiez les autorisations accordées par votre politique, puis choisissez Créer une politique pour enregistrer votre travail.

   La nouvelle politique s'affiche dans la liste des politiques gérées et est prête à être attachée.

Étape 2 : Attacher des politiques à votre groupe d'utilisateurs test

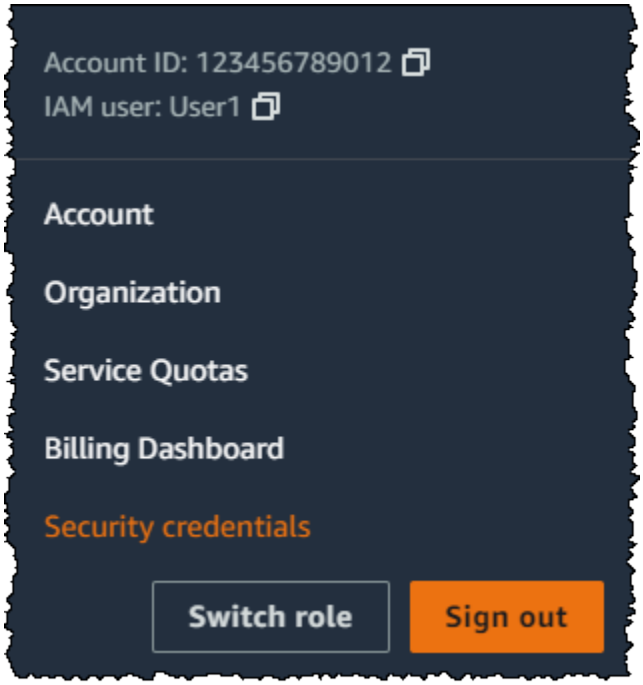
Vous allez maintenant attacher deux politiques à votre groupe d'utilisateurs IAM test. Elles seront utilisées pour octroyer des autorisations protégées par MFA.

1. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs).
2. Dans la zone de recherche, saisissez **`EC2MFA`**, puis sélectionnez le nom du groupe (et pas la case à cocher) dans la liste.
3. Choisissez l'onglet Autorisations, puis Ajouter des autorisations, et enfin Attacher des politiques.
4. Sur la page Attacher des politiques d'autorisation au groupe EC2 MFA, dans le champ de recherche, tapez. **`EC2Full`** Cochez ensuite la case à côté d'Amazon EC2 FullAccess dans la liste. N'enregistrez pas vos modifications pour le moment.
5. Dans la zone de recherche, saisissez **`Force`**, puis cochez la case située en regard de Force_MFA dans la liste.
6. Sélectionnez Attach Policies (Attacher des politiques).

Étape 3 : Test de l'accès utilisateur

Dans cette étape du didacticiel, vous vous connectez en tant qu'utilisateur test afin de vérifier que la politique fonctionne comme prévu.

1. Connectez-vous à votre annonce **`MFAUser`** avec Compte AWS le mot de passe que vous avez attribué dans la section précédente. Utilisez l'URL : `https://<alias or account ID number>.signin.aws.amazon.com/console`
2. Sélectionnez EC2 pour ouvrir la console Amazon EC2 et vérifiez que l'utilisateur ne dispose d'aucune autorisation.
3. Dans la barre de navigation en haut à droite, sélectionnez votre nom d'utilisateur `MFAUser`, puis Security Credentials (Informations d'identification de sécurité).



4. Maintenant, ajoutez un dispositif MFA. Dans la section Multi-Factor Authentication (MFA) (Authentification multi-facteurs (MFA)) sélectionnez Assign MFA device (Attribuer un dispositif MFA).

> **Note**
>
> Vous pouvez recevoir une erreur que vous n'êtes pas autorisé à effectuer `iam:DeleteVirtualMFADevice`. Cela peut se produire si quelqu'un a commencé précédemment à attribuer un dispositif MFA virtuel pour cet utilisateur et a annulé le processus. Pour continuer, vous ou un autre administrateur devez supprimer le dispositif MFA virtuel non attribué existant de l'utilisateur. Pour de plus amples informations, veuillez consulter Je ne suis pas autorisé à effectuer : iam : DeleteVirtualMFADevice.

5. Dans le cadre de ce didacticiel, nous utilisons un appareil MFA (basé sur un logiciel) virtuel, comme l'application Google Authenticator sur un téléphone portable. Choisissez l'application Authenticator, puis cliquez sur Next (Suivant).

   IAM génère et affiche les informations de configuration du dispositif MFA virtuel, notamment un graphique de code QR. Le graphique est une représentation de la clé de configuration secrète que l'on peut saisir manuellement sur des dispositifs qui ne prennent pas en charge les codes QR.

6. Ouvrez votre application MFA virtuelle. (Pour obtenir une liste des applications que vous pouvez utiliser pour héberger des dispositifs MFA virtuels, veuillez consulter Applications MFA virtuelles.) Si l'application MFA virtuelle prend en charge plusieurs comptes (plusieurs dispositifs MFA virtuels), sélectionnez l'option permettant de créer un compte (un nouveau dispositif MFA virtuel).

7. Déterminez si l'application MFA prend en charge les codes QR, puis effectuez l'une des actions suivantes :

   - Dans l'assistant, sélectionnez Show QR code (Afficher le code QR). Utilisez ensuite l'application pour analyser le code QR. Par exemple, vous pouvez choisir l'icône de caméra ou une option similaire à Scan code, puis utiliser la caméra du dispositif pour analyser le code.
   - Dans l'assistant Set up device (Configurer le dispositif), sélectionnez Show secret key (Afficher la clé secrète), puis saisissez la clé secrète dans votre application MFA.

   Une fois que vous avez terminé, le dispositif MFA virtuel commence à générer des mots de passe uniques.

8. Dans l'assistant Configurer le dispositif, dans la zone Saisir le code à partir de votre application d'authentification, saisissez le mot de passe unique qui s'affiche actuellement sur le dispositif MFA virtuel. Sélectionnez Register MFA (Enregistrer le dispositif MFA).

   > **Important**
   >
   > Envoyez votre demande immédiatement après avoir généré le code. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la requête, l'appareil MFA est associé avec succès à l'utilisateur. Cependant, l'appareil MFA n'est pas synchronisé. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période. Dans ce cas, vous pouvez resynchroniser le dispositif.

Le périphérique MFA virtuel est maintenant prêt à être utilisé avec. AWS

9. Déconnectez-vous de la console, puis reconnectez-vous en tant que **`MFAUser`**. Cette fois, vous AWS êtes invité à saisir un code MFA sur votre téléphone. Lorsque vous recevez le code, entrez-le dans le champ et sélectionnez Submit (Soumettre).
10. Sélectionnez EC2 pour rouvrir la console Amazon EC2. Notez que, cette fois, vous pouvez voir toutes les informations et effectuer les actions que vous souhaitez. Si vous accédez à une autre console en tant qu'utilisateur, les messages d'accès refusé s'affichent. La raison en est que les politiques de ce didacticiel n'accordent l'accès qu'à Amazon EC2.

Ressources connexes

Pour plus d'informations, veuillez consulter les rubriques suivantes :

- AWS Multi-factor authentification dans IAM
- Connexion compatible avec la MFA

Tutoriel IAM : utilisez un CloudFormation modèle pour créer un fournisseur d'identité SAML (IdP)

Pour configurer la fédération SAML pour votre AWS compte, vous devez créer un fournisseur d'identité SAML (IdP). Ce didacticiel explique comment utiliser un CloudFormation modèle pour créer un IdP SAML qui établit la confiance entre AWS et votre IdP externe.

Le modèle crée un IdP SAML configuré avec le document de métadonnées de votre IdP. Les rôles IAM fédérés peuvent ensuite référencer cet IdP pour permettre aux utilisateurs authentifiés de votre IdP externe d'accéder aux ressources. AWS

La ressource déployée consiste en un IdP SAML configuré avec le document de métadonnées de votre IdP et des paramètres de chiffrement facultatifs.

Conditions préalables

Le didacticiel présume que vous avez déjà ce qui suit en place :

- Python 3.6 ou version ultérieure installée sur votre ordinateur local pour exécuter la commande Python utilisée dans ce didacticiel afin de formater le fichier XML des métadonnées SAML de votre IdP.
- Un document de métadonnées SAML provenant de votre IdP externe enregistré sous forme de fichier XML.

Créez un IdP SAML à l'aide de CloudFormation

Pour créer l'IdP SAML, créez un CloudFormation modèle et utilisez-le pour créer une pile contenant la ressource IdP.

Création du modèle

Tout d'abord, créez le CloudFormation modèle.

1. Dans la Modèle section, choisissez l'icône de copie sur l'onglet JSON ou YAML pour copier le contenu du modèle.
2. Copiez le contenu du modèle dans un nouveau fichier.
3. Enregistrez le fichier au niveau local.

Créez la pile .

Ensuite, utilisez le modèle que vous avez enregistré pour provisionner une CloudFormation pile.

1. Ouvrez la CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.
2. Sur la page Piles, dans le menu Créer une pile, choisissez Avec de nouvelles ressources (standard).
3. Spécifiez un modèle :
    a. Sous Prérequis, choisissez Choisir un modèle existant.
    b. Sous Spécifier le modèle, sélectionnez Charger un modèle de fichier.
    c. Choisissez Choisir un fichier, accédez au fichier du modèle, puis sélectionnez-le.
    d. Choisissez Suivant.
4. Spécifiez les détails de la pile suivants :
    a. Saisissez le nom de la pile.

b. Pour IdentityProviderName, vous pouvez laisser ce champ vide pour générer automatiquement un nom basé sur le nom de la pile, ou saisir un nom personnalisé pour votre IdP SAML. Les noms personnalisés doivent contenir uniquement des caractères alphanumériques, des points, des traits de soulignement et des tirets.

c. En IdentityProviderSAMLMetadataDocument effet, vous devez formater votre fichier XML de métadonnées SAML sur une seule ligne avant de le coller dans ce champ. Cela est nécessaire car la CloudFormation console nécessite que le contenu XML soit formaté sur une seule ligne lorsqu'il est transmis via les paramètres de la console.

Utilisez la commande Python suivante pour reformater votre fichier XML :

```
python3 -c "import sys, re; content=open(sys.argv[1]).read(); print(re.sub(r'>
\s+<', '><', content.replace('\n', '').replace('\r', '').strip()))" saml-
metadata.xml
```

> **Note**
>
> Le document de métadonnées SAML de l'IdP doit être formaté en une seule ligne pour la saisie des paramètres de la console. La commande Python supprime les sauts de ligne et les espaces supplémentaires afin de créer le format requis tout en conservant l'intégralité du contenu et de la structure d'origine.

Copiez le résultat de la commande Python et collez-le dans le IdentityProviderSAMLMetadataDocument champ.

Exemple de document de métadonnées SAML formaté (abrégé) :

```
<?xml version="1.0" encoding="UTF-8"?><md:EntityDescriptor
 xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
 entityID="https://portal.sso.example.com/saml/assertion/
CompanyIdP"><md:IDPSSODescriptor WantAuthnRequestsSigned="false"
 protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"><md:KeyDescriptor
 use="signing"><ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/
xmldsig#"><ds:X509Data><ds:X509Certificate>MIIDXTCCAkWgAwIBAgIJAJC1HiIAZAiIMA0GCSqGSIb3
ds:X509Certificate></ds:X509Data></ds:KeyInfo></
md:KeyDescriptor><md:SingleLogoutService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
 Location="https://portal.sso.example.com/saml/logout/
```

```
CompanyIdP"/><md:NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent</md:NameIDFormat><md:SingleSignOnService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://
portal.sso.example.com/saml/assertion/CompanyIdP"/></md:IDPSSODescriptor></
md:EntityDescriptor>
```

d. Pour les autres paramètres, acceptez les valeurs par défaut ou saisissez les vôtres en fonction de vos besoins :

- IdentityProviderAddPrivateKey- Clé privée optionnelle pour déchiffrer les assertions SAML
- IdentityProviderAssertionEncryptionMode- Facultatif, définit le mode de cryptage pour les assertions SAML (autorisé, obligatoire ou vide)

e. Choisissez Suivant.

5. Configurez les options de pile :

a. Sous Options d'échec de pile, choisissez Supprimer toutes les ressources nouvellement créées.

> **Note**
>
> Le choix de cette option vous évite d'être facturé pour des ressources dont la politique de suppression indique qu'elles doivent être conservées même en cas d'échec de la création de la pile.

b. Acceptez toutes les autres valeurs par défaut.

c. Sous Fonctionnalités, cochez la case pour confirmer que des ressources IAM sont CloudFormation susceptibles de créer des ressources IAM sur votre compte.

d. Choisissez Suivant.

6. Vérifiez les détails de la pile et choisissez Envoyer.

CloudFormation crée la pile. Une fois la création de la pile terminée, les ressources de la pile sont prêtes à être utilisées. Vous pouvez utiliser l'onglet Ressources de la page détaillée de la pile pour afficher les ressources provisionnées dans votre compte.

La pile produira les valeurs suivantes, que vous pouvez consulter dans l'onglet Sorties :

- ProviderARN : l'ARN de l'IdP SAML créé (par exemple, `arn:aws:iam::123456789012:saml-provider/CompanyIdP`). Vous aurez besoin de cet ARN pour créer des rôles qui font confiance à ce fournisseur.
- ProviderName: nom de l'IdP SAML créé (par exemple, `CompanyIdP` si vous avez spécifié un nom personnalisé ou `my-saml-stack-saml-provider` si vous avez utilisé le nom par défaut).

Ces sorties sont également exportées, ce qui permet à d'autres piles CloudFormation de les importer à l'aide de la fonction `Fn::ImportValue`.

Vérifier l'IdP SAML

Une fois l'IdP SAML créé, vous pouvez vérifier sa configuration et noter son ARN à utiliser avec les rôles fédérés.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Identity providers (Fournisseurs d'identité).

   Vous devez voir votre IdP SAML nouvellement créé dans la liste.
3. Choisissez le nom de l'IdP pour en afficher les détails.

   Sur la page de détails de l'IdP, vous pouvez consulter le document de métadonnées SAML et d'autres détails de configuration.
4. Notez l'ARN du fournisseur affiché sur la page de détails.

   Vous aurez besoin de cet ARN pour créer des rôles IAM fédérés qui font confiance à cet IdP.
5. Vérifiez le document de métadonnées pour vous assurer qu'il correspond à ce que vous avez fourni par votre IdP externe.

Votre IdP SAML est désormais prêt à être utilisé par les rôles IAM fédérés. Vous pouvez créer des rôles qui font confiance à cet IdP pour permettre aux utilisateurs authentifiés de votre IdP externe d'assumer ces rôles et d'accéder aux ressources. AWS

Nettoyage : supprimez les ressources

Enfin, vous allez supprimer la pile et les ressources qu'elle contient.

1. Ouvrez la CloudFormation console.

2. Sur la page Piles, choisissez la pile créée à partir du modèle, choisissez Supprimer, puis confirmez Supprimer.

   CloudFormation lance la suppression de la pile et de toutes les ressources qu'elle inclut.

CloudFormation détails du modèle

Ressources

Le CloudFormation modèle de ce didacticiel créera la ressource suivante dans votre compte :

`AWS::IAM::SAMLProvider`: Un IdP SAML qui établit la confiance entre AWS et votre IdP externe.

Configuration

Le modèle comprend les paramètres configurables suivants :

- IdentityProviderName- Le nom de votre IdP SAML (laissez vide pour le nom généré automatiquement)

  Exemple : `CompanyIdP` ou `EnterpriseSSO`
- IdentityProviderSAMLMetadataDocument- Le document de métadonnées SAML de votre IdP externe (formaté sur une seule ligne)
- IdentityProviderAddPrivateKey- Clé privée optionnelle pour déchiffrer les assertions SAML
- IdentityProviderAssertionEncryptionMode- Facultatif, définit le mode de cryptage pour les assertions SAML

CloudFormation modèle

Enregistrez le code JSON ou YAML suivant dans un fichier distinct à utiliser comme CloudFormation modèle pour ce didacticiel.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "[AWSDocs] IAM: tutorial_saml-idp",
  "Parameters": {
    "IdentityProviderName": {
```

```
      "Type": "String",
      "Description": "Name of the SAML Identity Provider (leave empty for auto-
generated name like '{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[a-zA-Z0-9._-]+$",
      "ConstraintDescription": "Must be empty or contain only alphanumeric
 characters, periods, underscores, and hyphens"
    },
    "IdentityProviderSAMLMetadataDocument": {
      "Type": "String",
      "Description": "SAML metadata document from identity provider"
    },
    "IdentityProviderAddPrivateKey": {
      "Type": "String",
      "Description": "Optional private key for decrypting SAML assertions. The
 private key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm
 to decrypt SAML assertions.",
      "Default": ""
    },
    "IdentityProviderAssertionEncryptionMode": {
      "Type": "String",
      "Description": "Optional, sets encryption mode for SAML assertions",
      "Default": "",
      "AllowedValues": ["", "Allowed", "Required"]
    }
  },
  "Conditions": {
    "HasPrivateKey": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAddPrivateKey"}, ""]}]},
    "HasEncryptionMode": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAssertionEncryptionMode"}, ""]}]},
    "HasCustomName": {"Fn::Not": [{"Fn::Equals": [{"Ref": "IdentityProviderName"},
""]}]}
  },
  "Resources": {
    "SAMLProvider": {
      "Type": "AWS::IAM::SAMLProvider",
      "Properties": {
        "Name": {"Fn::If": ["HasCustomName", {"Ref": "IdentityProviderName"},
{"Ref": "AWS::NoValue"}]},
        "SamlMetadataDocument": {"Ref": "IdentityProviderSAMLMetadataDocument"},
        "Tags": [
          {
            "Key": "Name",
```

```
            "Value": {"Fn::If": ["HasCustomName", {"Ref": "IdentityProviderName"},
 {"Fn::Sub": "${AWS::StackName}-saml-provider"}]}
          }
        ],
        "AddPrivateKey": {"Fn::If": ["HasPrivateKey", {"Ref":
 "IdentityProviderAddPrivateKey"}, {"Ref": "AWS::NoValue"}]},
        "AssertionEncryptionMode": {"Fn::If": ["HasEncryptionMode", {"Ref":
 "IdentityProviderAssertionEncryptionMode"}, {"Ref": "AWS::NoValue"}]}
      }
    }
  },
  "Outputs": {
    "ProviderARN": {
      "Description": "ARN of the created SAML Identity Provider",
      "Value": {"Ref": "SAMLProvider"},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-ProviderARN"}
      }
    },
    "ProviderName": {
      "Description": "Name of the SAML Identity Provider",
      "Value": {"Fn::If": ["HasCustomName", {"Ref": "IdentityProviderName"},
 {"Fn::Sub": "${AWS::StackName}-saml-provider"}]},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-ProviderName"}
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: '[AWSDocs] IAM: tutorial_saml-idp'

Parameters:
  IdentityProviderName:
    Type: String
    Description: Name of the SAML Identity Provider (leave empty for auto-generated
 name like '{StackName}-{UniqueId}')
    Default: ""
    AllowedPattern: '^$|^[a-zA-Z0-9._-]+$'
```

```
    ConstraintDescription: 'Must be empty or contain only alphanumeric characters,
 periods, underscores, and hyphens'

  IdentityProviderSAMLMetadataDocument:
    Type: String
    Description: SAML metadata document from identity provider

  IdentityProviderAddPrivateKey:
    Type: String
    Description: Optional private key for decrypting SAML assertions. The private
 key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm to
 decrypt SAML assertions.
    Default: ""

  IdentityProviderAssertionEncryptionMode:
    Type: String
    Description: Optional, sets encryption mode for SAML assertions
    Default: ""
    AllowedValues:
      - ""
      - "Allowed"
      - "Required"

Conditions:
  HasPrivateKey: !Not [!Equals [!Ref IdentityProviderAddPrivateKey, ""]]
  HasEncryptionMode: !Not [!Equals [!Ref IdentityProviderAssertionEncryptionMode,
 ""]]
  HasCustomName: !Not [!Equals [!Ref IdentityProviderName, ""]]

Resources:
  SAMLProvider:
    Type: 'AWS::IAM::SAMLProvider'
    Properties:
      Name: !If
        - HasCustomName
        - !Ref IdentityProviderName
        - !Ref AWS::NoValue
      SamlMetadataDocument: !Ref IdentityProviderSAMLMetadataDocument
      Tags:
        - Key: Name
          Value: !If
            - HasCustomName
            - !Ref IdentityProviderName
            - !Sub '${AWS::StackName}-saml-provider'
```

```
      AddPrivateKey: !If
        - HasPrivateKey
        - !Ref IdentityProviderAddPrivateKey
        - !Ref AWS::NoValue
      AssertionEncryptionMode: !If
        - HasEncryptionMode
        - !Ref IdentityProviderAssertionEncryptionMode
        - !Ref AWS::NoValue

Outputs:
  ProviderARN:
    Description: 'ARN of the created SAML Identity Provider'
    Value: !Ref SAMLProvider
    Export:
      Name: !Sub '${AWS::StackName}-ProviderARN'

  ProviderName:
    Description: 'Name of the SAML Identity Provider'
    Value: !If
      - HasCustomName
      - !Ref IdentityProviderName
      - !Sub '${AWS::StackName}-saml-provider'
    Export:
      Name: !Sub '${AWS::StackName}-ProviderName'
```

Tutoriel IAM : utilisez un CloudFormation modèle pour créer un rôle IAM fédéré SAML

Lorsqu'un fournisseur d'identité (IdP) SAML est déjà configuré dans AWS votre compte, vous pouvez créer des rôles IAM fédérés qui font confiance à cet IdP. Ce didacticiel explique comment utiliser un CloudFormation modèle pour créer un rôle IAM fédéré SAML qui peut être assumé par des utilisateurs authentifiés via votre IdP externe.

Le modèle crée un rôle IAM fédéré avec une politique d'approbation permettant à votre IdP SAML d'endosser le rôle. Les utilisateurs authentifiés par votre IdP externe peuvent endosser ce rôle pour accéder aux ressources AWS en fonction des autorisations associées au rôle.

La ressource déployée est composée des éléments suivants :

- Un rôle IAM fédéré qui fait confiance à votre idP SAML existant.

- Des politiques gérées configurables qui peuvent être associées au rôle afin d'accorder des autorisations spécifiques.
- Des paramètres de limite d'autorisation et de durée de session facultatifs.

Conditions préalables

Le didacticiel présume que vous avez déjà ce qui suit en place :

- Un IdP SAML existant configuré dans AWS votre compte. Si vous n'en avez pas, vous pouvez en créer un à l'aide du didacticiel Tutoriel IAM : utilisez un CloudFormation modèle pour créer un fournisseur d'identité SAML (IdP).
- L'ARN de votre IdP SAML, que vous devrez spécifier comme paramètre lors de la création de la pile.
- Python 3.6 ou version ultérieure installée sur votre ordinateur local pour exécuter la commande Python utilisée dans ce didacticiel afin de formater le fichier XML des métadonnées SAML de votre IdP.

Créez un rôle fédéré SAML à l'aide de CloudFormation

Pour créer le rôle fédéré SAML, vous allez créer un CloudFormation modèle et l'utiliser pour créer une pile contenant le rôle.

Création du modèle

Créez d'abord le CloudFormation modèle.

1. Dans la section Modèle, cliquez sur l'icône de copie dans l'onglet JSON ou YAML pour copier le contenu du modèle.
2. Copiez le contenu du modèle dans un nouveau fichier.
3. Enregistrez le fichier au niveau local.

Créez la pile .

Ensuite, utilisez le modèle que vous avez enregistré pour approvisionner une CloudFormation pile.

1. Ouvrez la CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.

2. Sur la page Piles, dans le menu Créer une pile, choisissez Avec de nouvelles ressources (standard).
3. Spécifiez un modèle :
   a. Sous Prérequis, choisissez Choisir un modèle existant.
   b. Sous Spécifier le modèle, sélectionnez Charger un modèle de fichier.
   c. Choisissez Choisir un fichier, accédez au fichier du modèle, puis sélectionnez-le.
   d. Choisissez Suivant.
4. Spécifiez les détails de la pile suivants :
   a. Saisissez le nom de la pile.
   b. Pour SAMLProviderARN, saisissez l'ARN de votre IdP SAML existant. Cela doit être au format `arn:aws:iam::123456789012:saml-provider/YourProviderName`.

      Exemple : `arn:aws:iam::123456789012:saml-provider/CompanyIdP`

      > **Note**
      >
      > Si vous avez créé votre IdP SAML à l'aide Tutoriel IAM : utilisez un CloudFormation modèle pour créer un fournisseur d'identité SAML (IdP) du didacticiel, vous trouverez l'ARN du fournisseur dans l'onglet Sorties de CloudFormation cette pile.

   c. En RoleNameeffet, vous pouvez laisser ce champ vide pour générer automatiquement un nom basé sur le nom de la pile, ou saisir un nom personnalisé pour le rôle IAM.

      Exemple : `SAML-Developer-Access` ou `SAML-ReadOnly-Role`
   d. Pour les autres paramètres, acceptez les valeurs par défaut ou saisissez les vôtres en fonction de vos besoins :
      - RoleSessionDuration- Durée maximale de session en secondes (3600-43200, 7200 par défaut)

        Exemple : `14400` (4 heures)
      - RolePermissionsBoundary- ARN facultatif d'une politique de limite d'autorisations

        Exemple : `arn:aws:iam::123456789012:policy/DeveloperBoundary`
      - RolePath- Chemin pour le rôle IAM (la valeur par défaut est/)

Exemple : `/saml-roles/`

- ManagedPolicy1-5- ARN facultatifs contenant jusqu'à 5 politiques gérées à associer

  Exemple pour ManagedPolicy 1 : `arn:aws:iam::aws:policy/ReadOnlyAccess`

  Exemple pour ManagedPolicy 2 : `arn:aws:iam::123456789012:policy/CustomPolicy`

e. Choisissez Suivant.

5. Configurez les options de pile :

   a. Sous Options d'échec de pile, choisissez Supprimer toutes les ressources nouvellement créées.

   > **Note**
   >
   > Le choix de cette option vous évite d'être facturé pour des ressources dont la politique de suppression indique qu'elles doivent être conservées même en cas d'échec de la création de la pile.

   b. Acceptez toutes les autres valeurs par défaut.

   c. Sous Fonctionnalités, cochez la case pour confirmer que des ressources IAM CloudFormation peuvent être créées dans votre compte.

   d. Choisissez Suivant.

6. Vérifiez les détails de la pile et choisissez Envoyer.

CloudFormation crée la pile. Une fois la création de la pile terminée, les ressources de la pile sont prêtes à être utilisées. Vous pouvez utiliser l'onglet Ressources de la page détaillée de la pile pour afficher les ressources provisionnées dans votre compte.

La pile produira la valeur suivante, que vous pouvez consulter dans l'onglet Sorties :

- RoleARN : ARN du rôle IAM créé (par exemple, `arn:aws:iam::123456789012:role/SAML-Developer-Access` ou `arn:aws:iam::123456789012:role/stack-name-a1b2c3d4` si vous utilisez un nom généré automatiquement).

Vous aurez besoin de cet ARN de rôle lors de la configuration de votre IdP afin d'envoyer les attributs SAML appropriés pour l'attribution des rôles.

Tester le rôle fédéré SAML

Une fois le rôle fédéré SAML créé, vous pouvez vérifier sa configuration et tester la configuration de la fédération.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Rôles.
3. Trouvez et choisissez le rôle fédéré que vous venez de créer.

   Si vous avez fourni un nom de rôle personnalisé, recherchez ce nom. Si vous avez laissé le RoleName paramètre vide, le rôle aura un nom généré automatiquement en fonction du nom de la pile et d'un identifiant unique.
4. Choisissez l'onglet Relations de confiance pour consulter la politique de confiance.

   La politique de confiance doit indiquer que votre IdP SAML est fiable pour assumer ce rôle à condition que l'audience SAML (`SAML:aud`) corresponde à `https://signin.aws.amazon.com/saml`.
5. Sélectionnez l'onglet Autorisations pour consulter les politiques associées.

   Vous pouvez voir toutes les politiques gérées associées au rôle lors de sa création.
6. Notez l'ARN du rôle affiché sur la page de résumé du rôle.

   Vous aurez besoin de cet ARN pour configurer votre IdP externe afin de permettre aux utilisateurs d'assumer ce rôle.

Votre rôle fédéré SAML est maintenant prêt à être utilisé. Configurez votre IdP externe pour inclure l'ARN de ce rôle dans les assertions SAML, et les utilisateurs authentifiés pourront assumer ce rôle pour accéder aux ressources. AWS

Nettoyage : supprimez les ressources

Enfin, vous allez supprimer la pile et les ressources qu'elle contient.

1. Ouvrez la CloudFormation console.

2. Sur la page Piles, choisissez la pile créée à partir du modèle, choisissez Supprimer, puis confirmez Supprimer.

   CloudFormation initie la suppression de la pile et de toutes les ressources qu'elle contient.

CloudFormation détails du modèle

Ressources

Le CloudFormation modèle de ce didacticiel créera la ressource suivante dans votre compte :

- `AWS::IAM::Role` : un rôle IAM fédéré qui peut être endossé par les utilisateurs authentifiés via votre IdP SAML.

Configuration

Le modèle comprend les paramètres configurables suivants :

- RoleName- Nom du rôle IAM (laissez le champ vide pour le nom généré automatiquement)
- SAMLProviderARN : ARN de l'IdP SAML (obligatoire)
- RoleSessionDuration- Durée maximale de session en secondes (3600-43200, 7200 par défaut)
- RolePermissionsBoundary- ARN facultatif de la politique de limite des autorisations
- RolePath- Chemin du rôle IAM (par défaut/)
- ManagedPolicy1-5- ARN facultatifs contenant jusqu'à 5 politiques gérées à associer

CloudFormation modèle

Enregistrez le code JSON ou YAML suivant dans un fichier distinct à utiliser comme CloudFormation modèle pour ce didacticiel.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "[AWSDocs] IAM: tutorial_saml-federated-role",
  "Parameters": {
    "RoleName": {
      "Type": "String",
```

```
      "Description": "Name of the IAM Role (leave empty for auto-generated name like
'{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[\\w+=,.@-]{1,64}$",
      "ConstraintDescription": "Must be empty or 1-64 characters and can contain
alphanumeric characters and +=,.@-"
    },
    "SAMLProviderARN": {
      "Type": "String",
      "Description": "ARN of the SAML Identity Provider",
      "AllowedPattern": "^arn:aws:iam::\\d{12}:saml-provider/[a-zA-Z0-9._-]+$",
      "ConstraintDescription": "Must be a valid SAML provider ARN"
    },
    "RoleSessionDuration": {
      "Type": "Number",
      "Description": "The maximum session duration (in seconds) that you want to set
for the specified role (3600-43200)",
      "MinValue": 3600,
      "MaxValue": 43200,
      "Default": 7200
    },
    "RolePermissionsBoundary": {
      "Type": "String",
      "Description": "Optional ARN of the permissions boundary policy (leave empty
for none)",
      "Default": ""
    },
    "RolePath": {
      "Type": "String",
      "Description": "Path for the IAM role (must start and end with /)",
      "Default": "/",
      "AllowedPattern": "^\/.*\/$|^\/$",
      "ConstraintDescription": "Role path must start and end with forward slash (/)"
    },
    "RoleManagedPolicy1": {
      "Type": "String",
      "Description": "Optional managed policy ARN 1",
      "Default": ""
    },
    "RoleManagedPolicy2": {
      "Type": "String",
      "Description": "Optional managed policy ARN 2",
      "Default": ""
    },
```

```
    "RoleManagedPolicy3": {
      "Type": "String",
      "Description": "Optional managed policy ARN 3",
      "Default": ""
    },
    "RoleManagedPolicy4": {
      "Type": "String",
      "Description": "Optional managed policy ARN 4",
      "Default": ""
    },
    "RoleManagedPolicy5": {
      "Type": "String",
      "Description": "Optional managed policy ARN 5",
      "Default": ""
    }
  },
  "Conditions": {
    "HasCustomRoleName": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleName"}, ""]}]},
    "HasPermissionsBoundary": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"RolePermissionsBoundary"}, ""]}]},
    "HasPolicy1": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy1"},
""]}]},
    "HasPolicy2": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy2"},
""]}]},
    "HasPolicy3": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy3"},
""]}]},
    "HasPolicy4": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy4"},
""]}]},
    "HasPolicy5": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy5"}, ""]}]}
  },
  "Resources": {
    "SAMLFederatedRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "RoleName": {"Fn::If": ["HasCustomRoleName", {"Ref": "RoleName"}, {"Ref":
"AWS::NoValue"}]},
        "Description": "IAM role with SAML provider trust",
        "MaxSessionDuration": {"Ref": "RoleSessionDuration"},
        "PermissionsBoundary": {"Fn::If": ["HasPermissionsBoundary", {"Ref":
"RolePermissionsBoundary"}, {"Ref": "AWS::NoValue"}]},
        "Path": {"Ref": "RolePath"},
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
```

```
            {
              "Effect": "Allow",
              "Principal": {
                "Federated": {"Ref": "SAMLProviderARN"}
              },
              "Action": "sts:AssumeRoleWithSAML",
              "Condition": {
                "StringEquals": {
                  "SAML:aud": "https://signin.aws.amazon.com/saml"
                }
              }
            }
          ]
        },
        "ManagedPolicyArns": {
          "Fn::Split": [
            ",",
            {
              "Fn::Join": [
                ",",
                [
                  {"Fn::If": ["HasPolicy1", {"Ref": "RoleManagedPolicy1"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy2", {"Ref": "RoleManagedPolicy2"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy3", {"Ref": "RoleManagedPolicy3"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy4", {"Ref": "RoleManagedPolicy4"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy5", {"Ref": "RoleManagedPolicy5"}, {"Ref":
"AWS::NoValue"}]}
                ]
              ]
            }
          ]
        }
      }
    }
  },
  "Outputs": {
    "RoleARN": {
      "Description": "ARN of the created IAM Role",
      "Value": {"Fn::GetAtt": ["SAMLFederatedRole", "Arn"]},
      "Export": {
```

```
        "Name": {"Fn::Sub": "${AWS::StackName}-RoleARN"}
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: '[AWSDocs] IAM: tutorial_saml-federated-role'

Parameters:
  RoleName:
    Type: String
    Description: 'Name of the IAM Role (leave empty for auto-generated name like
 ''{StackName}-{UniqueId}'')'
    Default: ""
    AllowedPattern: '^$|^[\w+=,.@-]{1,64}$'
    ConstraintDescription: 'Must be empty or 1-64 characters and can contain
 alphanumeric characters and +=,.@-'

  SAMLProviderARN:
    Type: String
    Description: 'ARN of the SAML Identity Provider'
    AllowedPattern: '^arn:aws:iam::\d{12}:saml-provider/[a-zA-Z0-9._-]+$'
    ConstraintDescription: 'Must be a valid SAML provider ARN'

  RoleSessionDuration:
    Type: Number
    Description: 'The maximum session duration (in seconds) that you want to set for
 the specified role (3600-43200)'
    MinValue: 3600
    MaxValue: 43200
    Default: 7200

  RolePermissionsBoundary:
    Type: String
    Description: Optional ARN of the permissions boundary policy (leave empty for
 none)
    Default: ""

  RolePath:
    Type: String
```

```
    Description: 'Path for the IAM role (must start and end with /)'
    Default: "/"
    AllowedPattern: '^\/.*\/$|^\/$'
    ConstraintDescription: 'Role path must start and end with forward slash (/)'

  RoleManagedPolicy1:
    Type: String
    Description: Optional managed policy ARN 1
    Default: ""
  RoleManagedPolicy2:
    Type: String
    Description: Optional managed policy ARN 2
    Default: ""
  RoleManagedPolicy3:
    Type: String
    Description: Optional managed policy ARN 3
    Default: ""
  RoleManagedPolicy4:
    Type: String
    Description: Optional managed policy ARN 4
    Default: ""
  RoleManagedPolicy5:
    Type: String
    Description: Optional managed policy ARN 5
    Default: ""

Conditions:
  HasCustomRoleName: !Not [!Equals [!Ref RoleName, ""]]
  HasPermissionsBoundary: !Not [!Equals [!Ref RolePermissionsBoundary, ""]]
  HasPolicy1: !Not [!Equals [!Ref RoleManagedPolicy1, ""]]
  HasPolicy2: !Not [!Equals [!Ref RoleManagedPolicy2, ""]]
  HasPolicy3: !Not [!Equals [!Ref RoleManagedPolicy3, ""]]
  HasPolicy4: !Not [!Equals [!Ref RoleManagedPolicy4, ""]]
  HasPolicy5: !Not [!Equals [!Ref RoleManagedPolicy5, ""]]

Resources:
  SAMLFederatedRole:
    Type: 'AWS::IAM::Role'
    Properties:
      RoleName: !If
        - HasCustomRoleName
        - !Ref RoleName
        - !Ref AWS::NoValue
      Description: 'IAM role with SAML provider trust'
```

```
      MaxSessionDuration: !Ref RoleSessionDuration
      PermissionsBoundary: !If
        - HasPermissionsBoundary
        - !Ref RolePermissionsBoundary
        - !Ref AWS::NoValue
      Path: !Ref RolePath
      AssumeRolePolicyDocument:
        Version: '2012-10-17       '
        Statement:
          - Effect: Allow
            Principal:
              Federated: !Ref SAMLProviderARN
            Action: 'sts:AssumeRoleWithSAML'
            Condition:
              StringEquals:
                'SAML:aud': 'https://signin.aws.amazon.com/saml'
      ManagedPolicyArns:
        !Split
          - ','
          - !Join
            - ','
            - - !If [HasPolicy1, !Ref RoleManagedPolicy1, !Ref 'AWS::NoValue']
              - !If [HasPolicy2, !Ref RoleManagedPolicy2, !Ref 'AWS::NoValue']
              - !If [HasPolicy3, !Ref RoleManagedPolicy3, !Ref 'AWS::NoValue']
              - !If [HasPolicy4, !Ref RoleManagedPolicy4, !Ref 'AWS::NoValue']
              - !If [HasPolicy5, !Ref RoleManagedPolicy5, !Ref 'AWS::NoValue']

Outputs:
  RoleARN:
    Description: 'ARN of the created IAM Role'
    Value: !GetAtt SAMLFederatedRole.Arn
    Export:
      Name: !Sub '${AWS::StackName}-RoleARN'
```

Tutoriel IAM : utilisez un CloudFormation modèle pour créer un fournisseur d'identité SAML (IdP) et un rôle IAM fédéré SAML

Pour vous familiariser avec la fédération SAML et ses fonctionnalités, vous allez utiliser un CloudFormation modèle pour configurer un fournisseur d'identité SAML (IdP) et le rôle IAM fédéré associé. Ce didacticiel vous montre comment créer les deux ressources ensemble dans une seule pile.

Le modèle crée un IdP SAML qui peut être utilisé pour un accès fédéré aux AWS ressources, ainsi qu'un rôle IAM qui fait confiance au fournisseur SAML. Les utilisateurs authentifiés par votre IdP externe peuvent assumer ce rôle pour accéder AWS aux ressources.

Les ressources déployées sont composées des éléments suivants :

- Un IdP SAML configuré avec le document de métadonnées de votre IdP.
- Un rôle IAM fédéré qui fait confiance à l'IdP SAML et peut être endossé par des utilisateurs authentifiés.
- Des politiques gérées configurables qui peuvent être associées au rôle afin d'accorder des autorisations spécifiques.

Conditions préalables

Le didacticiel présume que vous avez déjà ce qui suit en place :

- Python 3.6 ou version ultérieure installée sur votre ordinateur local pour exécuter la commande Python utilisée dans ce didacticiel afin de formater le fichier XML des métadonnées SAML de votre IdP.
- Un document de métadonnées SAML provenant de votre IdP externe enregistré sous forme de fichier XML.

Créez un IdP et un rôle SAML à l'aide de CloudFormation

Pour créer l'IdP SAML et le rôle fédéré, vous allez créer un CloudFormation modèle et l'utiliser pour créer une pile contenant les deux ressources.

Création du modèle

Tout d'abord, créez le CloudFormation modèle.

1. Dans la section Modèle, cliquez sur l'icône de copie dans l'onglet JSON ou YAML pour copier le contenu du modèle.
2. Copiez le contenu du modèle dans un nouveau fichier.
3. Enregistrez le fichier au niveau local.

Créez la pile .

Ensuite, utilisez le modèle que vous avez enregistré pour provisionner une CloudFormation pile.

1. Ouvrez la CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.
2. Sur la page Piles, dans le menu Créer une pile, choisissez Avec de nouvelles ressources (standard).
3. Spécifiez un modèle :
   a. Sous Prérequis, choisissez Choisir un modèle existant.
   b. Sous Spécifier le modèle, sélectionnez Charger un modèle de fichier.
   c. Choisissez Choisir un fichier, accédez au fichier du modèle, puis sélectionnez-le.
   d. Choisissez Suivant.
4. Spécifiez les détails de la pile suivants :
   a. Saisissez le nom de la pile.
   b. Pour IdentityProviderName, vous pouvez laisser ce champ vide pour générer automatiquement un nom basé sur le nom de la pile, ou saisir un nom personnalisé pour votre IdP SAML.

      Exemple : `CompanyIdP` ou `EnterpriseSSO`
   c. En IdentityProviderSAMLMetadataDocument effet, vous devez formater votre fichier XML de métadonnées SAML sur une seule ligne avant de le coller dans ce champ. Cela est nécessaire car la CloudFormation console nécessite que le contenu XML soit formaté sur une seule ligne lorsqu'il est transmis via les paramètres de la console.

      Utilisez la commande Python suivante pour reformater votre fichier XML :

```
python3 -c "import sys, re; content=open(sys.argv[1]).read(); print(re.sub(r'>\s+<', '><', content.replace('\n', '').replace('\r', '').strip()))" saml-metadata.xml
```

> **Note**
>
> Le document de métadonnées SAML de l'IdP doit être formaté en une seule ligne pour la saisie des paramètres de la console. La commande Python supprime les

sauts de ligne et les espaces supplémentaires afin de créer le format requis tout en conservant l'intégralité du contenu et de la structure d'origine.

Copiez le résultat de la commande Python et collez-le dans le IdentityProviderSAMLMetadataDocument champ.

Exemple de document de métadonnées SAML formaté (abrégé) :

```
<?xml version="1.0" encoding="UTF-8"?><md:EntityDescriptor
 xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
 entityID="https://portal.sso.example.com/saml/assertion/
CompanyIdP"><md:IDPSSODescriptor WantAuthnRequestsSigned="false"
 protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"><md:KeyDescriptor
 use="signing"><ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/
xmldsig#"><ds:X509Data><ds:X509Certificate>MIIDXTCCAkWgAwIBAgIJAJC1HiIAZAiIMA0GCSqGSIb3
ds:X509Certificate></ds:X509Data></ds:KeyInfo></
md:KeyDescriptor><md:SingleLogoutService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
 Location="https://portal.sso.example.com/saml/logout/
CompanyIdP"/><md:NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent</md:NameIDFormat><md:SingleSignOnService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://
portal.sso.example.com/saml/assertion/CompanyIdP"/></md:IDPSSODescriptor></
md:EntityDescriptor>
```

d. Pour RoleName, vous pouvez laisser ce champ vide pour générer automatiquement un nom basé sur le nom de la pile, ou saisir un nom personnalisé pour le rôle IAM fédéré.

   Exemple : `SAML-Developer-Access` ou `SAML-ReadOnly-Role`

e. Pour les autres paramètres, acceptez les valeurs par défaut ou saisissez les vôtres en fonction de vos besoins :

   - IdentityProviderAddPrivateKey- Clé privée optionnelle pour déchiffrer les assertions SAML
   - IdentityProviderAssertionEncryptionMode- Mode de cryptage pour les assertions SAML

     Exemples de valeurs : `Allowed`, `Required`, ou laissez vide pour aucun chiffrement
   - RoleSessionDuration- Durée maximale de session en secondes (3600-43200, par défaut 7200)

Exemple : `14400` (4 heures)

- RolePermissionsBoundary- ARN facultatif d'une politique de limites d'autorisations

  Exemple : `arn:aws:iam::123456789012:policy/DeveloperBoundary`

- RolePath- Chemin du rôle IAM (la valeur par défaut est/)

  Exemple : `/saml-roles/`

- RoleManagedPolicy1-5- Des ARN facultatifs contenant jusqu'à 5 politiques gérées à associer

  Exemple pour RoleManagedPolicy1 : `arn:aws:iam::aws:policy/ReadOnlyAccess`

  Exemple pour RoleManagedPolicy2 : `arn:aws:iam::123456789012:policy/CustomPolicy`

f. Choisissez Suivant.

5. Configurez les options de pile :

   a. Sous Options d'échec de pile, choisissez Supprimer toutes les ressources nouvellement créées.

   > **Note**
   >
   > Le choix de cette option vous évite d'être facturé pour des ressources dont la politique de suppression indique qu'elles doivent être conservées même en cas d'échec de la création de la pile.

   b. Acceptez toutes les autres valeurs par défaut.

   c. Sous Fonctionnalités, cochez la case pour confirmer que des ressources IAM sont CloudFormation susceptibles de créer des ressources IAM sur votre compte.

   d. Choisissez Suivant.

6. Vérifiez les détails de la pile et choisissez Envoyer.

CloudFormation crée la pile. Une fois la création de la pile terminée, les ressources de la pile sont prêtes à être utilisées. Vous pouvez utiliser l'onglet Ressources de la page détaillée de la pile pour afficher les ressources provisionnées dans votre compte.

La pile produira les valeurs suivantes, que vous pouvez consulter dans l'onglet Sorties :

- RoleARN : ARN du rôle IAM créé (par exemple, `arn:aws:iam::123456789012:role/SAML-Developer-Access` ou `arn:aws:iam::123456789012:role/stack-name-a1b2c3d4` si vous utilisez un nom généré automatiquement).
- IdentityProviderARN : ARN de l'IdP SAML créé (par exemple,`arn:aws:iam::123456789012:saml-provider/CompanyIdP`).

Vous aurez besoin de ces deux ARN lors de la configuration de votre IdP afin d'envoyer les attributs SAML appropriés pour l'attribution des rôles.

Tester la fédération SAML

Une fois que l'IdP SAML et le rôle fédéré ont été créés, vous pouvez tester la configuration de la fédération.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Identity providers (Fournisseurs d'identité).

   Vous devez voir votre IdP SAML nouvellement créé dans la liste.
3. Choisissez le nom de l'IdP pour en afficher les détails.

   Sur la page de détails de l'IdP, vous pouvez consulter le document de métadonnées SAML et d'autres détails de configuration.
4. Dans le panneau de navigation, choisissez Rôles.
5. Trouvez et choisissez le rôle fédéré que vous venez de créer.

   Sur la page de détails du rôle, vous pouvez voir la politique de confiance qui permet à l'IdP SAML d'endosser ce rôle.
6. Choisissez l'onglet Relations de confiance pour consulter la politique de confiance.

   La politique de confiance doit indiquer que l'IdP SAML est fiable pour assumer ce rôle à condition que l'audience SAML (`SAML:aud`) corresponde à `https://signin.aws.amazon.com/saml`.

Nettoyage : supprimez les ressources

Enfin, vous allez supprimer la pile et les ressources qu'elle contient.

1. Ouvrez la CloudFormation console.
2. Sur la page Piles, choisissez la pile créée à partir du modèle, choisissez Supprimer, puis confirmez Supprimer.

   CloudFormation lance la suppression de la pile et de toutes les ressources qu'elle inclut.

CloudFormation détails du modèle

Ressources

Le CloudFormation modèle de ce didacticiel créera les ressources suivantes dans votre compte :

- `AWS::IAM::SAMLProvider`: Un IdP SAML qui établit la confiance entre AWS et votre IdP externe.
- `AWS::IAM::Role` : un rôle IAM fédéré qui peut être endossé par les utilisateurs authentifiés via l'IdP SAML.

Configuration

Le modèle comprend les paramètres configurables suivants :

- IdentityProviderName- Nom de l'IdP SAML (laissez ce champ vide pour le nom généré automatiquement)
- IdentityProviderSAMLMetadataDocument- Document de métadonnées SAML provenant de votre IdP (obligatoire)
- IdentityProviderAddPrivateKey- Clé privée optionnelle pour déchiffrer les assertions SAML
- IdentityProviderAssertionEncryptionMode- Mode de cryptage pour les assertions SAML
- RoleName- Nom du rôle IAM (laissez ce champ vide pour le nom généré automatiquement)
- RolePath- Chemin du rôle IAM (par défaut/)
- RolePermissionsBoundary- ARN facultatif de la politique de limites d'autorisations
- RoleSessionDuration- Durée maximale de session en secondes (3600-43200, par défaut 7200)
- RoleManagedPolicy1-5- Des ARN facultatifs contenant jusqu'à 5 politiques gérées à associer

CloudFormation modèle

Enregistrez le code JSON ou YAML suivant dans un fichier distinct à utiliser comme CloudFormation modèle pour ce didacticiel.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "[AWSDocs] IAM: tutorial_saml-idp-and-federated-role",
  "Parameters": {
    "IdentityProviderName": {
      "Type": "String",
      "Description": "Name of the SAML Identity Provider (leave empty for auto-
generated name like '{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[a-zA-Z0-9._-]+$",
      "ConstraintDescription": "Must be empty or contain only alphanumeric
 characters, periods, underscores, and hyphens"
    },
    "IdentityProviderSAMLMetadataDocument": {
      "Type": "String",
      "Description": "SAML metadata document from identity provider"
    },
    "IdentityProviderAddPrivateKey": {
      "Type": "String",
      "Description": "Optional private key for decrypting SAML assertions. The
 private key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm
 to decrypt SAML assertions.",
      "Default": ""
    },
    "IdentityProviderAssertionEncryptionMode": {
      "Type": "String",
      "Description": "Optional, sets encryption mode for SAML assertions",
      "Default": "",
      "AllowedValues": ["", "Allowed", "Required"]
    },
    "RoleName": {
      "Type": "String",
      "Description": "Name of the IAM Role (leave empty for auto-generated name like
 '{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[\\w+=,.@-]{1,64}$",
```

```
      "ConstraintDescription": "Must be empty or 1-64 characters and can contain
alphanumeric characters and +=,.@-"
    },
    "RolePath": {
      "Type": "String",
      "Description": "Path for the IAM Role",
      "AllowedPattern": "(^\\/$)|(^\\/.*\\/$)",
      "Default": "/"
    },
    "RolePermissionsBoundary": {
      "Type": "String",
      "Description": "Optional ARN of the permissions boundary policy (leave empty
for none)",
      "Default": ""
    },
    "RoleSessionDuration": {
      "Description": "The maximum session duration (in seconds) that you want to set
for the specified role (3600-43200)",
      "Type": "Number",
      "MinValue": 3600,
      "MaxValue": 43200,
      "Default": 7200
    },
    "RoleManagedPolicy1": {
      "Type": "String",
      "Description": "Optional managed policy ARN 1",
      "Default": ""
    },
    "RoleManagedPolicy2": {
      "Type": "String",
      "Description": "Optional managed policy ARN 2",
      "Default": ""
    },
    "RoleManagedPolicy3": {
      "Type": "String",
      "Description": "Optional managed policy ARN 3",
      "Default": ""
    },
    "RoleManagedPolicy4": {
      "Type": "String",
      "Description": "Optional managed policy ARN 4",
      "Default": ""
    },
    "RoleManagedPolicy5": {
```

```
      "Type": "String",
      "Description": "Optional managed policy ARN 5",
      "Default": ""
    }
  },
  "Conditions": {
    "HasCustomProviderName": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderName"}, ""]}]},
    "HasCustomRoleName": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleName"}, ""]}]},
    "HasPermissionsBoundary": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"RolePermissionsBoundary"}, ""]}]},
    "HasPolicy1": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy1"},
""]}]},
    "HasPolicy2": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy2"},
""]}]},
    "HasPolicy3": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy3"},
""]}]},
    "HasPolicy4": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy4"},
""]}]},
    "HasPolicy5": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy5"},
""]}]},
    "HasPrivateKey": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAddPrivateKey"}, ""]}]},
    "HasAssertionEncryptionMode": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAssertionEncryptionMode"}, ""]}]}
  },
  "Resources": {
    "SAMLProvider": {
      "Type": "AWS::IAM::SAMLProvider",
      "Properties": {
        "Name": {"Fn::If": ["HasCustomProviderName", {"Ref":
"IdentityProviderName"}, {"Ref": "AWS::NoValue"}]},
        "SamlMetadataDocument": {"Ref": "IdentityProviderSAMLMetadataDocument"},
        "AddPrivateKey": {"Fn::If": ["HasPrivateKey", {"Ref":
"IdentityProviderAddPrivateKey"}, {"Ref": "AWS::NoValue"}]},
        "AssertionEncryptionMode": {"Fn::If": ["HasAssertionEncryptionMode", {"Ref":
"IdentityProviderAssertionEncryptionMode"}, {"Ref": "AWS::NoValue"}]}
      }
    },
    "SAMLFederatedRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "RoleName": {"Fn::If": ["HasCustomRoleName", {"Ref": "RoleName"}, {"Ref":
"AWS::NoValue"}]},
```

```
        "Path": {"Ref": "RolePath"},
        "Description": "SAML federated IAM role for SSO access with specified
permissions",
        "MaxSessionDuration": {"Ref": "RoleSessionDuration"},
        "PermissionsBoundary": {"Fn::If": ["HasPermissionsBoundary", {"Ref":
"RolePermissionsBoundary"}, {"Ref": "AWS::NoValue"}]},
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Federated": {"Ref": "SAMLProvider"}
              },
              "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity",
                "sts:TagSession"
              ],
              "Condition": {
                "StringEquals": {
                  "SAML:aud": "https://signin.aws.amazon.com/saml"
                }
              }
            }
          ]
        },
        "ManagedPolicyArns": {
          "Fn::Split": [
            ",",
            {
              "Fn::Join": [
                ",",
                [
                  {"Fn::If": ["HasPolicy1", {"Ref": "RoleManagedPolicy1"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy2", {"Ref": "RoleManagedPolicy2"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy3", {"Ref": "RoleManagedPolicy3"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy4", {"Ref": "RoleManagedPolicy4"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy5", {"Ref": "RoleManagedPolicy5"}, {"Ref":
"AWS::NoValue"}]}
```

```
                ]
              ]
            }
          ]
        }
      }
    }
  },
  "Outputs": {
    "RoleARN": {
      "Description": "ARN of the created IAM Role",
      "Value": {"Fn::GetAtt": ["SAMLFederatedRole", "Arn"]},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-RoleARN"}
      }
    },
    "IdentityProviderARN": {
      "Description": "ARN of the created SAML Identity Provider",
      "Value": {"Ref": "SAMLProvider"},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-IdentityProviderARN"}
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: '[AWSDocs] IAM: tutorial_saml-idp-and-federated-role'

Parameters:
  IdentityProviderName:
    Type: String
    Description: Name of the SAML Identity Provider (leave empty for auto-generated
 name like '{StackName}-{UniqueId}')
    Default: ""
    AllowedPattern: '^$|^[a-zA-Z0-9._-]+$'
    ConstraintDescription: Must be empty or contain only alphanumeric characters,
 periods, underscores, and hyphens

  IdentityProviderSAMLMetadataDocument:
    Type: String
```

```
    Description: SAML metadata document from identity provider

  IdentityProviderAddPrivateKey:
    Type: String
    Description: Optional private key for decrypting SAML assertions. The private
key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm to
decrypt SAML assertions.
    Default: ""

  IdentityProviderAssertionEncryptionMode:
    Type: String
    Description: Optional, sets encryption mode for SAML assertions
    Default: ""
    AllowedValues:
      - ""
      - "Allowed"
      - "Required"

  RoleName:
    Type: String
    Description: Name of the IAM Role (leave empty for auto-generated name like
'{StackName}-{UniqueId}')
    Default: ""
    AllowedPattern: '^$|^[\w+=,.@-]{1,64}$'
    ConstraintDescription: "Must be empty or 1-64 characters and can contain
alphanumeric characters and +=,.@-"

  RolePath:
    Type: String
    Description: Path for the IAM Role
    AllowedPattern: (^\/$)|(^\/.*\/$)
    Default: "/"

  RolePermissionsBoundary:
    Type: String
    Description: Optional ARN of the permissions boundary policy (leave empty for
none)
    Default: ""

  RoleSessionDuration:
    Description: The maximum session duration (in seconds) that you want to set for
the specified role (3600-43200)
    Type: Number
    MinValue: 3600
```

```
    MaxValue: 43200
    Default: 7200

  RoleManagedPolicy1:
    Type: String
    Description: Optional managed policy ARN 1
    Default: ""
  RoleManagedPolicy2:
    Type: String
    Description: Optional managed policy ARN 2
    Default: ""
  RoleManagedPolicy3:
    Type: String
    Description: Optional managed policy ARN 3
    Default: ""
  RoleManagedPolicy4:
    Type: String
    Description: Optional managed policy ARN 4
    Default: ""
  RoleManagedPolicy5:
    Type: String
    Description: Optional managed policy ARN 5
    Default: ""

Conditions:
  HasCustomProviderName: !Not [!Equals [!Ref IdentityProviderName, ""]]
  HasCustomRoleName: !Not [!Equals [!Ref RoleName, ""]]
  HasPermissionsBoundary: !Not [!Equals [!Ref RolePermissionsBoundary, ""]]
  HasPolicy1: !Not [!Equals [!Ref RoleManagedPolicy1, ""]]
  HasPolicy2: !Not [!Equals [!Ref RoleManagedPolicy2, ""]]
  HasPolicy3: !Not [!Equals [!Ref RoleManagedPolicy3, ""]]
  HasPolicy4: !Not [!Equals [!Ref RoleManagedPolicy4, ""]]
  HasPolicy5: !Not [!Equals [!Ref RoleManagedPolicy5, ""]]
  HasPrivateKey: !Not [!Equals [!Ref IdentityProviderAddPrivateKey, ""]]
  HasAssertionEncryptionMode: !Not [!Equals [!Ref
 IdentityProviderAssertionEncryptionMode, ""]]

Resources:
  SAMLProvider:
    Type: AWS::IAM::SAMLProvider
    Properties:
      Name: !If
        - HasCustomProviderName
        - !Ref IdentityProviderName
```

```
        - !Ref AWS::NoValue
      SamlMetadataDocument: !Ref IdentityProviderSAMLMetadataDocument
      AddPrivateKey: !If
        - HasPrivateKey
        - !Ref IdentityProviderAddPrivateKey
        - !Ref AWS::NoValue
      AssertionEncryptionMode: !If
        - HasAssertionEncryptionMode
        - !Ref IdentityProviderAssertionEncryptionMode
        - !Ref AWS::NoValue

  SAMLFederatedRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !If
        - HasCustomRoleName
        - !Ref RoleName
        - !Ref AWS::NoValue
      Path: !Ref RolePath
      Description: "SAML federated IAM role for SSO access with specified
permissions"
      MaxSessionDuration: !Ref RoleSessionDuration
      PermissionsBoundary: !If
        - HasPermissionsBoundary
        - !Ref RolePermissionsBoundary
        - !Ref AWS::NoValue
      AssumeRolePolicyDocument:
        Version: '2012-10-17       '
        Statement:
          - Effect: Allow
            Principal:
              Federated: !Ref SAMLProvider
            Action:
              - 'sts:AssumeRole'
              - 'sts:SetSourceIdentity'
              - 'sts:TagSession'
            Condition:
              StringEquals:
                'SAML:aud': 'https://signin.aws.amazon.com/saml'
      ManagedPolicyArns: !Split
        - ','
        - !Join
          - ','
          - - !If [HasPolicy1, !Ref RoleManagedPolicy1, !Ref "AWS::NoValue"]
```

```
            - !If [HasPolicy2, !Ref RoleManagedPolicy2, !Ref "AWS::NoValue"]
            - !If [HasPolicy3, !Ref RoleManagedPolicy3, !Ref "AWS::NoValue"]
            - !If [HasPolicy4, !Ref RoleManagedPolicy4, !Ref "AWS::NoValue"]
            - !If [HasPolicy5, !Ref RoleManagedPolicy5, !Ref "AWS::NoValue"]

Outputs:
  RoleARN:
    Description: ARN of the created IAM Role
    Value: !GetAtt SAMLFederatedRole.Arn
    Export:
      Name: !Sub '${AWS::StackName}-RoleARN'

  IdentityProviderARN:
    Description: ARN of the created SAML Identity Provider
    Value: !Ref SAMLProvider
    Export:
      Name: !Sub '${AWS::StackName}-IdentityProviderARN'
```

IAM identités

Une identité IAM peut être associée à une ou plusieurs politiques, qui déterminent les actions qu'une identité est autorisée à effectuer, sur quelles AWS ressources et dans quelles conditions. Les identités IAM incluent les utilisateurs IAM, les groupes IAM et les rôles IAM. Une identité IAM est un type d'entité qui représente un utilisateur humain ou une charge de travail programmatique qui peut être authentifié et autorisé à effectuer des actions dans Comptes AWS. Les entités IAM comprennent les utilisateurs IAM et les rôles IAM. Pour les définitions des termes couramment utilisés, consultez Termes.

Vous pouvez fédérer des identités existantes à partir d'un fournisseur d'identité externe. Ces identités assumeront des rôles IAM pour accéder aux AWS ressources. Pour de plus amples informations, veuillez consulter the section called "Fournisseurs d'identité et fédération au sein AWS".

Vous pouvez également l'utiliser AWS IAM Identity Center pour créer et gérer les identités et l'accès aux AWS ressources. Les ensembles d'autorisations IAM Identity Center créent automatiquement les rôles IAM nécessaires pour fournir l'accès aux ressources. Pour plus d'informations, consultez En quoi consiste IAM Identity Center ?.

Vous pouvez utiliser le gestionnaire d'accès aux comptes, une fonctionnalité IAM qui vous permet d'attribuer des rôles IAM existants avec l'ensemble complet des fonctionnalités de rôles IAM aux utilisateurs et aux groupes IAM Identity Center. Vous pouvez l'utiliser conjointement avec des ensembles d'autorisations ou seul.

Utilisateur racine d'un compte AWS Il s'agit d'un Compte AWS principal qui est créé lors de votre Compte AWS création. L'utilisateur root a accès à tous les AWS services et ressources du compte. Pour de plus amples informations, veuillez consulter the section called "Utilisateur racine IAM".

> **Note**
>
> - Suivez les Pratiques exemplaires en matière de sécurité dans IAM lorsque vous travaillez avec des identités IAM.
> - Suivez les meilleures pratiques applicables à l'utilisateur racine de votre Compte AWS lorsque vous travaillez avec l'utilisateur racine.
> - Si vous ne parvenez pas à vous connecter, consultez Se connecter à l' Console de gestion AWS.

Utilisateur racine IAM

Lorsque vous créez un compte pour la première fois Compte AWS, vous commencez avec une seule identité de connexion qui dispose d'un accès complet à toutes Services AWS les ressources du compte. Cette identité est appelée utilisateur Compte AWS root. Pour plus d’informations, consultez Utilisateur racine du compte AWS.

Utilisateurs IAM

Un utilisateur IAM est une identité au sein de vous Compte AWS qui possède des autorisations spécifiques pour une seule personne ou une seule application. Pour plus d’informations, consultez Utilisateurs IAM.

Groupes d'utilisateurs IAM

Un groupes d’utilisateurs IAM est une identité qui spécifie un ensemble d’utilisateurs IAM. Pour plus d’informations, consultez la section Groupes d’utilisateurs.

Rôles IAM

Un rôle IAM est une identité au sein de vous Compte AWS qui possède des autorisations spécifiques. S’il est comparable à un utilisateur IAM, il n’est toutefois pas associé à une personne déterminée. Pour en savoir plus, consultez Rôles IAM.

Utilisateur racine d'un compte AWS

Lorsque vous créez un compte Amazon Web Services (AWS) pour la première fois, vous utilisez une identité de connexion unique qui donne un accès complet à tous les AWS services et ressources du compte. Cette identité est appelée utilisateur root du AWS compte. L'adresse e-mail et le mot de passe que vous avez utilisés pour créer le vôtre Compte AWS sont les informations d'identification que vous utilisez pour vous connecter en tant qu'utilisateur root.

- Utilisez l’utilisateur racine pour effectuer les tâches qui nécessitent des autorisations de niveau racine. Pour obtenir la liste complète des tâches qui nécessitent que vous vous connectiez en tant qu'utilisateur root, veuillez consulter Tâches nécessitant les informations d'identification de l'utilisateur root.
- Suivez les Meilleures pratiques d’utilisateur racine pour votre Compte AWS.

- Si vous ne parvenez pas à vous connecter, consultez Se connecter à l' Console de gestion AWS.

> **Important**
>
> Nous vous recommandons vivement de ne pas utiliser l'utilisateur root pour vos tâches quotidiennes et de suivre les bonnes pratiques de l'utilisateur root pour votre Compte AWS. Protégez vos informations d'identification d'utilisateur racine et utilisez-les pour effectuer les tâches que seul l'utilisateur racine peut effectuer. Pour obtenir la liste complète des tâches qui nécessitent que vous vous connectiez en tant qu'utilisateur root, veuillez consulter Tâches nécessitant les informations d'identification de l'utilisateur root.

Bien que l'authentification multifactorielle (MFA) soit appliquée par défaut pour les utilisateurs racine, elle nécessite une action de la part du client pour être ajoutée lors de la création initiale du compte ou à la demande lors de la connexion. Pour de plus amples informations sur l'utilisation de l'authentification multifactorielle (MFA) pour protéger l'utilisateur racine, veuillez consulter Multi-factor authentification pour Utilisateur racine d'un compte AWS.

Outre l'accès à la console, vous pouvez gérer l'accès programmatique pour l'utilisateur root. Vous le pouvez Création de clés d'accès pour l'utilisateur racine et Suppression d'une clé d'accès de l'utilisateur racine quand vous n'en avez plus besoin.

Gestion centralisée de l'accès root pour les comptes membres

Pour vous aider à gérer les informations d'identification à grande échelle, vous pouvez sécuriser de manière centralisée l'accès aux informations d'identification des utilisateurs racine pour les comptes membres dans AWS Organizations. Lorsque vous l'activez AWS Organizations, vous regroupez tous vos AWS comptes au sein d'une organisation pour une gestion centralisée. La centralisation de l'accès racine vous permet de supprimer les informations d'identification de l'utilisateur racine et d'effectuer les tâches privilégiées suivantes sur les comptes membres.

Suppression des informations d'identification de l'utilisateur racine des comptes membres

Après avoir centralisé l'accès racine pour les comptes membres, vous pouvez choisir de supprimer les informations d'identification de l'utilisateur racine des comptes membres de votre Organizations. Vous pouvez supprimer le mot de passe d'utilisateur racine, les clés d'accès, les certificats de signature, ainsi que désactiver l'authentification multifactorielle (MFA). Les nouveaux comptes que vous créez dans Organizations ne disposent par défaut d'aucunes informations

d’identification d’utilisateur racine. Les comptes membres ne peuvent pas se connecter à leur utilisateur racine ni récupérer le mot de passe de leur utilisateur racine si la récupération de compte n’est pas activée.

Effectuez les tâches privilégiées nécessitant les informations d’identification de l’utilisateur racine

Certaines tâches ne peuvent être effectuées que lorsque vous vous connectez en tant qu’utilisateur racine d’un compte. Certaines d’entre elles Tâches nécessitant les informations d'identification de l'utilisateur root peuvent être effectuées par le compte de gestion ou par l’administrateur délégué d’IAM. Pour en savoir plus sur les actions privilégiées sur les comptes membres, consultez Exécuter une tâche privilégiée.

Activer la restauration du compte de l’utilisateur racine

Si vous devez récupérer les informations d’identification de l’utilisateur racine pour un compte membre, le compte de gestion des Organizations ou l’administrateur délégué peut exécuter la tâche privilégiée Autoriser la récupération du mot de passe. La personne ayant accès à la boîte de réception de l’utilisateur racine pour le compte membre peut réinitialiser le mot de passe de l’utilisateur racine pour récupérer les informations d’identification de l’utilisateur racine. Nous vous recommandons de supprimer les informations d'identification de l'utilisateur root une fois que vous avez terminé la tâche nécessitant l'accès à l'utilisateur root.

Centralisation de l’accès racine pour les comptes membres

Les informations d'identification de l'utilisateur root sont les informations d'identification initiales attribuées à chacun Compte AWS ayant un accès complet à tous les AWS services et ressources du compte. Lorsque vous activez cette option AWS Organizations, vous regroupez tous vos AWS comptes au sein d'une organisation pour une gestion centralisée. Chaque compte membre dispose de son propre utilisateur racine avec des autorisations par défaut lui permettant d’effectuer toute action dans le compte membre. Nous vous recommandons de sécuriser de manière centralisée les informations d'identification de l'utilisateur root en cas d'utilisation Comptes AWS gérée afin AWS Organizations d'empêcher la récupération des informations d'identification de l'utilisateur root et l'accès à grande échelle.

Après avoir centralisé l’accès racine, vous pouvez choisir de supprimer les informations d’identification de l’utilisateur racine des comptes membres de votre organisation. Vous pouvez supprimer le mot de passe d’utilisateur racine, les clés d’accès, les certificats de signature, ainsi que désactiver l’authentification multifactorielle (MFA). Les nouveaux comptes que vous créez dans AWS Organizations ne disposent par défaut d’aucune information d’identification d’utilisateur racine.

Les comptes membres ne peuvent pas se connecter à leur utilisateur racine ni récupérer le mot de passe de leur utilisateur racine.

> **Note**
>
> Certaines tâches Tâches nécessitant les informations d'identification de l'utilisateur root peuvent être effectuées par le compte de gestion ou l'administrateur délégué pour IAM, mais d'autres ne peuvent être effectuées que lorsque vous vous connectez en tant qu'utilisateur racine d'un compte.
> Si vous devez récupérer les informations d'identification de l'utilisateur racine pour un compte membre pour exécuter l'une de ces tâches, suivez les étapes décrites dans Exécuter une tâche privilégiée et sélectionnez Autoriser la récupération du mot de passe. La personne ayant accès à la boîte de réception de l'utilisateur racine pour le compte membre peut alors suivre les étapes pour réinitialiser le mot de passe de l'utilisateur racine et se connecter à l'utilisateur racine du compte membre.
> Nous vous recommandons de supprimer les informations d'identification de l'utilisateur racine une fois que vous avez terminé la tâche nécessitant l'accès à l'utilisateur racine.

Conditions préalables

Avant de centraliser l'accès racine, vous devez configurer un compte avec les paramètres suivants :

- Vous devez détenir les autorisations IAM suivants :
  - `iam:GetAccessKeyLastUsed`
  - `iam:GetAccountSummary`
  - `iam:GetLoginProfile`
  - `iam:GetUser`
  - `iam:ListAccessKeys`
  - `iam:ListMFADevices`
  - `iam:ListSigningCertificates`
  - `sts:AssumeRoot`

> **Note**
>
> Pour auditer l'état des informations d'identification de l'utilisateur root d'un compte membre, vous pouvez utiliser la politique IAMAuditRootUserCredentials AWS gérée pour limiter les autorisations lorsque vous effectuez une tâche privilégiée sur un compte AWS Organizations membre, ou utiliser n'importe quelle politique ayant accès à`iam:GetAccountSummary`.
> Pour générer le rapport sur les informations d'identification de l'utilisateur racine, les autres politiques ont uniquement besoin de l'action `iam:GetAccountSummary` pour produire le même résultat. Vous pouvez également répertorier ou obtenir les informations d'identification de chaque utilisateur racine, notamment :
>
> - Si un mot de passe utilisateur racine est présent
> - Si une clé d'accès utilisateur racine est présente et quand elle a été utilisée pour la dernière fois
> - Si l'utilisateur racine possède des certificats de signature associés
> - Dispositifs MFA associés à l'utilisateur racine
> - Liste du statut consolidé des informations d'identification de l'utilisateur racine

- Vous devez gérer votre Comptes AWS entrée AWS Organizations.
- Pour activer cette fonctionnalité de votre organisation, vous devez disposer de l'autorisation suivante :
  - `iam:EnableOrganizationsRootCredentialsManagement`
  - `iam:EnableOrganizationsRootSessions`
  - `iam:ListOrganizationsFeatures`
  - `organizations:EnableAwsServiceAccess`
  - `organizations:ListAccountsForParent`
  - `organizations:RegisterDelegatedAdministrator`
- Pour garantir un fonctionnement optimal de la console, nous vous recommandons d'activer les autorisations supplémentaires suivantes :
  - `organizations:DescribeAccount`
  - `organizations:DescribeOrganization`
  - `organizations:ListAWSServiceAccessForOrganization`

- `organizations:ListDelegatedAdministrators`
- `organizations:ListOrganizationalUnitsForParent`
- `organizations:ListParents`
- `organizations:ListTagsForResource`

Activation de l'accès racine centralisé (console)

Pour activer cette fonctionnalité pour les comptes de membres dans Console de gestion AWS

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le volet de navigation de la console, choisissez Gestion de l'accès racine, puis sélectionnez Activer.

   > **Note**
   >
   > Si vous voyez que la gestion de l'accès root est désactivée, activez l'accès sécurisé pour Gestion des identités et des accès AWS entrer AWS Organizations. Pour plus d'informations, consultez AWS IAM et AWS Organizations dans le Guide de l'utilisateur AWS Organizations .

3. Dans la section Fonctionnalités à activer, choisissez les fonctionnalités à activer.

   - Sélectionnez Gestion des informations d'identification racine pour permettre au compte de gestion et à l'administrateur délégué d'IAM de supprimer les informations d'identification de l'utilisateur racine pour les comptes membres. Vous devez activer les actions racines privilégiées dans les comptes membres pour leur permettre de récupérer leurs informations d'identification d'utilisateur racine après leur suppression.
   - Sélectionnez Actions racine privilégiées dans les comptes membres pour permettre au compte de gestion et à l'administrateur délégué d'IAM d'effectuer certaines tâches nécessitant des informations d'identification de l'utilisateur racine.

4. (Facultatif) Saisissez l'ID de compte de l'administrateur délégué autorisé à gérer l'accès des utilisateurs racine et à effectuer des actions privilégiées sur les comptes membres. Nous recommandons un compte destiné à des fins de sécurité ou de gestion.
5. Sélectionnez Activer.

Activation de l'accès root centralisé (AWS CLI)

Pour activer un accès root centralisé depuis AWS Command Line Interface (AWS CLI)

1. Si vous n'avez pas encore activé l'accès sécurisé pour Gestion des identités et des accès AWS in AWS Organizations, utilisez la commande suivante : aws organizations enable-aws-service-access.
2. Utilisez la commande suivante pour autoriser le compte de gestion et l'administrateur délégué à supprimer les informations d'identification de l'utilisateur racine pour les comptes membres : aws iam enable-organizations-root-credentials-management.
3. Utilisez la commande suivante pour permettre au compte de gestion et à l'administrateur délégué d'effectuer certaines tâches nécessitant des informations d'identification de l'utilisateur racine : aws iam enable-organizations-root-sessions.
4. (Facultatif) Utilisez la commande suivante pour enregistrer un administrateur délégué : aws organizations register-delegated-administrator.

   L'exemple suivant affecte le compte 111111111111 en tant qu'administrateur délégué pour le service IAM.

   ```
   aws organizations register-delegated-administrator
   --service-principal iam.amazonaws.com
   --account-id 111111111111
   ```

Activation de l'accès root centralisé (AWS API)

Pour activer un accès root centralisé depuis AWS API

1. Si vous n'avez pas encore activé l'accès sécurisé pour Gestion des identités et des accès AWS in AWS Organizations, utilisez la commande suivante : EnableAWSServiceAccess.
2. Utilisez la commande suivante pour autoriser le compte de gestion et l'administrateur délégué à supprimer les informations d'identification de l'utilisateur root pour les comptes membres : EnableOrganizationsRootCredentialsManagement.
3. Utilisez la commande suivante pour autoriser le compte de gestion et l'administrateur délégué à effectuer certaines tâches nécessitant des informations d'identification utilisateur root : EnableOrganizationsRootSessions.

4. (Facultatif) Utilisez la commande suivante pour enregistrer un administrateur délégué : RegisterDelegatedAdministrator.

Étapes suivantes

Une fois que vous avez sécurisé de manière centralisée les informations d'identification privilégiées pour les comptes membres de votre organisation, consultez Exécuter une tâche privilégiée pour effectuer des actions privilégiées sur un compte membre.

Effectuez une tâche privilégiée sur un AWS Organizations compte membre

Le compte AWS Organizations de gestion ou un compte d'administrateur délégué pour IAM peut effectuer certaines tâches privilégiées sur les comptes membres qui nécessiteraient autrement des informations d'identification utilisateur root. Avec un accès root centralisé, ces tâches sont effectuées par le biais de sessions privilégiées de courte durée. Ces sessions fournissent des informations d'identification temporaires limitées à des actions privilégiées spécifiques, sans qu'il soit nécessaire que l'utilisateur root se connecte au compte du membre.

Une fois que vous avez lancé une session privilégiée, vous pouvez supprimer une politique de compartiment Amazon S3 mal configurée, supprimer une politique de file d'attente Amazon SQS mal configurée, supprimer les informations d'identification de l'utilisateur racine pour un compte membre et réactiver les informations d'identification de l'utilisateur racine pour un compte membre.

> **Note**
>
> Pour utiliser l'accès root centralisé, vous devez vous connecter via un compte de gestion ou un compte d'administrateur délégué en tant qu'utilisateur ou rôle IAM avec l'`sts:AssumeRoot`autorisation explicitement accordée. Vous ne pouvez pas utiliser les informations d'identification de l'utilisateur root pour appeler`sts:AssumeRoot`.

Conditions préalables

Avant de pouvoir lancer une session privilégiée, vous devez disposer des paramètres suivants :

- Vous avez activé l'accès racine centralisé dans votre organisation. Pour savoir comment activer cette fonctionnalité, consultez Centralisation de l'accès racine pour les comptes membres.
- Votre compte de gestion ou votre compte d'administrateur délégué dispose des autorisations suivantes : `sts:AssumeRoot`

Effectuer une action privilégiée sur un compte membre (console)

Pour lancer une session pour une action privilégiée dans un compte membre dans le Console de gestion AWS

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le volet de navigation de la console, choisissez Gestion de l'accès racine.
3. Sélectionnez un nom dans la liste des comptes membres, puis choisissez Effectuer une action privilégiée.
4. Choisissez l'action privilégiée que vous souhaitez effectuer dans le compte membre.
   - Sélectionnez Supprimer la politique de compartiment Amazon S3 pour supprimer une politique de compartiment mal configurée qui empêche tous les principaux d'accéder au compartiment Amazon S3.
     a. Choisissez Naviguer dans S3 pour sélectionner un nom parmi les compartiments appartenant au compte membre, puis sélectionnez Choisir.
     b. Choisissez Supprimer la politique de compartiment.
     c. Utilisez la console Amazon S3 pour corriger la politique de compartiment après avoir supprimé la politique mal configurée. Pour plus d'informations, consultez Ajout d'une politique de compartiment à l'aide de la console Amazon S3 dans le Guide de l'utilisateur Amazon S3.
   - Sélectionnez Supprimer la politique Amazon SQS pour supprimer une politique Amazon Simple Queue Service qui refuse à tous les principaux l'accès à une file d'attente Amazon SQS.
     a. Saisissez le nom de la file d'attente dans le champ Nom de la file d'attente SQS, puis sélectionnez Supprimer la politique SQS.
     b. Utilisez la console Amazon SQS pour corriger la politique de la file d'attente après avoir supprimé la politique mal configurée. Pour plus d'informations, consultez Configuration de stratégie d'accès dans Amazon SQS dans le Guide du développeur Amazon SQS.
   - Sélectionnez Supprimer les informations d'identification racine pour supprimer l'accès racine d'un compte membre. La suppression des informations d'identification de l'utilisateur racine supprime le mot de passe de l'utilisateur racine, les clés d'accès, les certificats de signature et désactive l'authentification multifactorielle (MFA) pour le compte membre.

    - Choisissez Supprimer les informations d'identification racine.
- Sélectionnez Autoriser la récupération du mot de passe pour récupérer les informations d'identification de l'utilisateur racine pour un compte membre.

  Cette option est disponible uniquement lorsque le compte membre ne possède pas d'informations d'identification d'utilisateur racine.

  a. Choisissez Autoriser la récupération du mot de passe.

  b. Après avoir effectué cette action privilégiée, la personne ayant accès à la boîte de réception de l'utilisateur racine pour le compte membre peut réinitialiser le mot de passe de l'utilisateur racine et se connecter à l'utilisateur racine du compte membre.

Effectuer une action privilégiée sur un compte membre (AWS CLI)

Pour lancer une session pour une action privilégiée dans un compte membre depuis le AWS Command Line Interface

1. Utilisez la commande suivante pour utiliser une session utilisateur racine : aws sts assume-root.

   > **Note**
   >
   > Le point de terminaison global n'est pas pris en charge pour `sts:AssumeRoot`. Vous devez envoyer cette demande à un point de AWS STS terminaison régional. Pour de plus amples informations, veuillez consulter Gérer AWS STS dans un Région AWS.

   Lorsque vous lancez une session utilisateur racine privilégiée pour un compte membre, vous devez définir `task-policy-arn` permettant d'étendre la session à l'action privilégiée à effectuer au cours de la session. Vous pouvez utiliser l'une des politiques gérées AWS suivantes pour définir les actions de session privilégiées.

   - IAMAuditRootUserCredentials
   - IAMCreateRootUserPassword
   - IAMDeleteRootUserCredentials
   - S3UnlockBucketPolicy
   - SQSUnlockQueuePolicy

Pour limiter les actions qu'un compte de gestion ou un administrateur délégué peut effectuer au cours d'une session utilisateur root privilégiée, vous pouvez utiliser la clé de AWS STS condition sts : TaskPolicyArn.

Dans l'exemple suivant, l'administrateur délégué suppose que root supprime les informations d'identification de l'utilisateur root pour l'ID du compte membre*111122223333*.

```
aws sts assume-root \
  --target-principal 111122223333 \
  --task-policy-arn arn=arn:aws:iam::aws:policy/root-task/
IAMDeleteRootUserCredentials \
  --duration-seconds 900
```

2. Utilisez `SessionToken`, `AccessKeyId` et `SecretAccessKey` depuis la réponse pour effectuer des actions privilégiées dans le compte membre. Vous pouvez omettre le nom d'utilisateur et le mot de passe dans la requête afin d'utiliser par défaut le compte membre.
   - Vérifiez l'état des informations d'identification de l'utilisateur racine. Utilisez les commandes suivantes pour vérifier l'état des informations d'identification de l'utilisateur racine pour un compte membre.
     - get-user
     - get-login-profile
     - list-access-keys
     - list-signing-certificates
     - list-mfa-devices
     - get-access-key-last-used
   - Supprimez les informations d'identification de l'utilisateur racine. Utilisez les commandes suivantes pour supprimer l'accès racine. Vous pouvez supprimer le mot de passe d'utilisateur racine, les clés d'accès, les certificats de signature, ainsi que désactiver l'authentification multifactorielle (MFA) pour supprimer tout accès et toute récupération de l'utilisateur racine.
     - delete-login-profile
     - delete-access-key
     - delete-signing-certificate

    - deactivate-mfa-device
- Supprimez le compartiment Amazon S3. Utilisez les commandes suivantes pour lire, modifier et supprimer une politique de compartiment mal configurée qui empêche tous les principaux d'accéder au compartiment Amazon S3.
    - list-buckets
    - get-bucket-policy
    - put-bucket-policy
    - delete-bucket-policy
- Supprimez la politique Amazon SQS. Utilisez les commandes suivantes pour afficher et supprimer une politique Amazon Simple Queue Service qui refuse à tous les principaux l'accès à une file d'attente Amazon SQS.
    - list-queues
    - get-queue-url
    - get-queue-attributes
    - set-queue-attributes
- Autorisez la récupération du mot de passe. Utilisez les commandes suivantes pour afficher le nom d'utilisateur et récupérer les informations d'identification de l'utilisateur racine pour un compte membre.
    - get-login-profile
    - create-login-profile

Effectuer une action privilégiée sur un compte membre (AWS API)

Pour lancer une session pour une action privilégiée dans un compte membre depuis le AWS API

1. Utilisez la commande suivante pour supposer une session utilisateur root : AssumeRoot.

> **Note**
>
> Le point de terminaison global n'est pas pris en charge pour AssumeRoot. Vous devez envoyer cette demande à un point de AWS STS terminaison régional. Pour de plus amples informations, veuillez consulter Gérer AWS STS dans un Région AWS.

Lorsque vous lancez une session utilisateur racine privilégiée pour un compte membre, vous devez définir `TaskPolicyArn` permettant d'étendre la session à l'action privilégiée à effectuer au cours de la session. Vous pouvez utiliser l'une des politiques AWS gérées suivantes pour définir les actions de session privilégiées.

- IAMAuditRootUserCredentials
- IAMCreateRootUserPassword
- IAMDeleteRootUserCredentials
- S3UnlockBucketPolicy
- SQSUnlockQueuePolicy

Pour limiter les actions qu'un compte de gestion ou un administrateur délégué peut effectuer au cours d'une session utilisateur root privilégiée, vous pouvez utiliser la clé de AWS STS condition sts : TaskPolicyArn.

Dans l'exemple suivant, l'administrateur délégué suppose que root peut lire, modifier et supprimer une politique basée sur les ressources mal configurée pour un compartiment Amazon S3 pour l'ID de compte membre. *111122223333*

```
https://sts.us-east-2.amazonaws.com/
  ?Version=2011-06-15
  &Action=AssumeRoot
  &TargetPrincipal=111122223333
  &PolicyArns.arn=arn:aws:iam::aws:policy/root-task/S3UnlockBucketPolicy
  &DurationSeconds=900
```

2. Utilisez `SessionToken`, `AccessKeyId` et `SecretAccessKey` depuis la réponse pour effectuer des actions privilégiées dans le compte membre. Vous pouvez omettre le nom d'utilisateur et le mot de passe dans la requête afin d'utiliser par défaut le compte membre.
   - Vérifiez l'état des informations d'identification de l'utilisateur racine. Utilisez les commandes suivantes pour vérifier l'état des informations d'identification de l'utilisateur racine pour un compte membre.
     - GetUser
     - GetLoginProfile
     - ListAccessKeys

- ListSigningCertificates
- ListMFADevices
- GetAccessKeyLastUsed

- Supprimez les informations d'identification de l'utilisateur racine. Utilisez les commandes suivantes pour supprimer l'accès racine. Vous pouvez supprimer le mot de passe d'utilisateur racine, les clés d'accès, les certificats de signature, ainsi que désactiver l'authentification multifactorielle (MFA) pour supprimer tout accès et toute récupération de l'utilisateur racine.

  - DeleteLoginProfile
  - DeleteAccessKey
  - DeleteSigningCertificate
  - DeactivateMfaDevice

- Supprimez le compartiment Amazon S3. Utilisez les commandes suivantes pour lire, modifier et supprimer une politique de compartiment mal configurée qui empêche tous les principaux d'accéder au compartiment Amazon S3.

  - ListBuckets
  - GetBucketPolicy
  - PutBucketPolicy
  - DeleteBucketPolicy

- Supprimez la politique Amazon SQS. Utilisez les commandes suivantes pour afficher et supprimer une politique Amazon Simple Queue Service qui refuse à tous les principaux l'accès à une file d'attente Amazon SQS.

  - ListQueues
  - GetQueueUrl
  - GetQueueAttributes
  - SetQueueAttributes

- Autorisez la récupération du mot de passe. Utilisez les commandes suivantes pour afficher le nom d'utilisateur et récupérer les informations d'identification de l'utilisateur racine pour un compte membre.

  - GetLoginProfile

- CreateLoginProfile

Multi-factor authentification pour Utilisateur racine d'un compte AWS

> **Important**
>
> AWS vous recommande d'utiliser un mot de passe ou une clé de sécurité pour le MFA AWS, dans la mesure du possible, car ils sont plus résistants aux attaques telles que le phishing. Pour de plus amples informations, veuillez consulter Clés d'accès et clés de sécurité.

Multi-factor L'authentification (MFA) est un mécanisme simple et efficace pour renforcer votre sécurité. Le premier facteur, votre mot de passe, est un secret que vous mémorisez, également appelé facteur de connaissance. Les autres facteurs peuvent être des facteurs liés à la possession (quelque chose que vous possédez, comme une clé de sécurité) ou des facteurs inhérents (quelque chose que vous êtes, comme un scan biométrique). Pour une sécurité accrue, nous vous recommandons vivement de configurer l'authentification multifacteur (MFA) afin de protéger vos AWS ressources.

> **Note**
>
> Tous les Compte AWS types (comptes autonomes, de gestion et comptes membres) nécessitent la configuration MFA pour leur utilisateur root. Les utilisateurs doivent enregistrer le MFA dans les 35 jours suivant leur première tentative de connexion pour accéder au MFA Console de gestion AWS si ce n'est pas déjà fait.

Vous pouvez activer l'authentification multifacteur pour les utilisateurs Utilisateur racine d'un compte AWS et IAM. Lorsque vous activez la MFA pour l'utilisateur racine, cela n'affecte que les informations d'identification de l'utilisateur racine. Pour en savoir plus sur l'activation de la MFA pour les utilisateurs IAM, consultez AWS Multi-factor authentification dans IAM.

> **Note**
>
> Comptes AWS managed using AWS Organizations peut avoir la possibilité de gérer de manière centralisée l'accès root pour les comptes membres afin d'empêcher la récupération des informations d'identification et l'accès à grande échelle. Si cette option est activée,

vous pouvez supprimer les informations d'identification de l'utilisateur racine des comptes membres, y compris les mots de passe et l'authentification multifactorielle (MFA), empêchant ainsi efficacement la connexion en tant qu'utilisateur racine, la récupération du mot de passe ou la configuration de la MFA. Si vous préférez conserver les méthodes de connexion par mot de passe, sécurisez votre compte en enregistrant l'authentification multifactorielle (MFA) afin de renforcer la protection de votre compte.

Avant d'activer la MFA pour votre utilisateur racine, vérifiez et mettez à jour les paramètres de votre compte et les informations de contact pour vous assurer que vous avez accès à l'adresse e-mail et au numéro de téléphone. Si votre dispositif MFA est perdu, volé ou ne fonctionne pas, vous pouvez toujours vous connecter en tant qu'utilisateur racine en vérifiant votre identité à l'aide de cet e-mail et de ce numéro de téléphone. Pour en savoir plus sur la connexion à l'aide d'autres facteurs d'authentification, consultez Restauration d'une identité protégée par MFA dans IAM. Pour désactiver cette fonction, contactez AWS Support.

AWS prend en charge les types d'authentification multifacteur suivants pour votre utilisateur root :

- Clés d'accès et clés de sécurité
- Applications d'authentification virtuelle
- Jetons TOTP matériels

Clés d'accès et clés de sécurité

Gestion des identités et des accès AWS prend en charge les clés d'accès et les clés de sécurité pour le MFA. Selon les normes FIDO, les clés d'accès utilisent la cryptographie à clé publique pour fournir une authentification robuste et résistante au phishing, qui est plus sûre que les mots de passe. AWS prend en charge deux types de clés d'accès : les clés d'accès liées à l'appareil (clés de sécurité) et les clés d'accès synchronisées.

- Clés de sécurité : il s'agit de périphériques physiques YubiKey, comme un, utilisés comme second facteur d'authentification. Une clé de sécurité unique peut prendre en charge plusieurs comptes utilisateur racine et utilisateurs IAM.
- Clés d'accès synchronisées : elles utilisent des gestionnaires d'informations d'identification provenant de fournisseurs tels que Google, Apple, Microsoft et de services tiers tels que 1Password, Dashlane et Bitwarden comme deuxième facteur.

Vous pouvez utiliser des authentificateurs biométriques intégrés, tels que Touch ID sur Apple MacBooks, pour déverrouiller votre gestionnaire d'informations d'identification et vous y connecter. AWS Les clés d'accès sont créées avec le fournisseur de votre choix à l'aide de votre empreinte digitale, de votre visage ou du code PIN de votre appareil. Vous pouvez utiliser une clé d'accès d'authentification entre appareils (CDA) provenant d'un appareil, comme un appareil mobile ou une clé de sécurité matérielle, pour vous connecter sur un autre appareil, tel qu'un ordinateur portable. Pour plus d'informations, consultez Authentification entre appareils (CDA).

Vous pouvez synchroniser les clés d'accès sur tous vos appareils pour faciliter les connexions et améliorer ainsi la facilité d'utilisation et la AWS capacité de restauration. Pour plus d'informations sur l'activation des clés d'accès et des clés de sécurité, consultez Activation d'une clé d'accès ou d'une clé de sécurité pour l'utilisateur racine (console).

La FIDO Alliance tient à jour une liste de tous les produits certifiés FIDO qui sont compatibles avec les spécifications FIDO.

Applications d'authentification virtuelle

Une application d'authentification virtuelle s'exécute sur un téléphone ou un autre dispositif et émule un dispositif physique. Les applications d'authentification virtuelle mettent en œuvre l'algorithme TOTP (mot de passe unique à durée limitée) et prennent en charge plusieurs jetons sur un seul dispositif. L'utilisateur doit saisir un code valide à partir du dispositif lorsqu'il y est invité lors de la connexion. Chaque jeton attribué à un utilisateur doit être unique. Un utilisateur ne peut pas saisir un code provenant du jeton d'un autre utilisateur pour s'authentifier.

Nous vous recommandons d'utiliser un dispositif MFA virtuel pendant l'attente de l'approbation d'achat du matériel ou pendant que vous attendez de recevoir votre matériel. Pour obtenir la liste de quelques applications prises en charge que vous pouvez utiliser en tant que périphériques MFA virtuels, consultez Multi-Factor Authentification (MFA). Pour obtenir des instructions sur la configuration d'un périphérique MFA virtuel avec AWS, consultezActivation d'un dispositif MFA virtuel pour l'utilisateur racine (console).

Jetons TOTP matériels

Un dispositif matériel qui génère un code numérique à six chiffres basé sur l'algorithme TOTP (mot de passe unique à durée limitée). L'utilisateur doit saisir un code valide à partir du dispositif sur une deuxième page web lors de la connexion. Chaque dispositif MFA attribué à un utilisateur doit être unique. Un utilisateur ne peut pas saisir un code à partir du périphérique d'un autre utilisateur pour s'authentifier. Pour plus d'informations sur les périphériques MFA matériels pris en charge, voir Multi-

Factor Authentification (MFA). Pour plus d'informations sur la configuration d'un jeton TOTP matériel avec AWS, consultez Activation d'un jeton TOTP matériel pour l'utilisateur root de l' (console).

Si vous souhaitez utiliser un dispositif MFA physique, nous vous recommandons d’utiliser les clés de sécurité FIDO comme alternative aux périphériques TOTP matériels. Les clés de sécurité FIDO offrent les avantages de ne pas nécessiter de batterie, de résister au hameçonnage et de prendre en charge plusieurs utilisateurs racine et utilisateurs IAM sur un seul appareil pour une sécurité renforcée.

Rubriques

- Activation d’une clé d’accès ou d’une clé de sécurité pour l’utilisateur racine (console)
- Activation d’un dispositif MFA virtuel pour l’utilisateur racine (console)
- Activation d'un jeton TOTP matériel pour l'utilisateur root de l' (console)

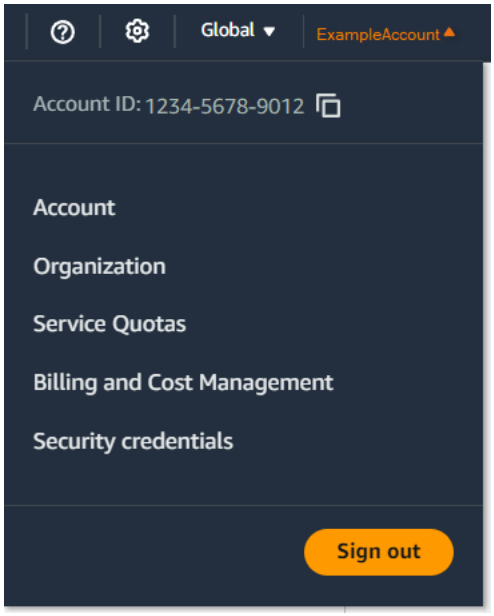
Activation d’une clé d’accès ou d’une clé de sécurité pour l’utilisateur racine (console)

Vous pouvez configurer et activer une clé d'accès pour votre utilisateur root Console de gestion AWS uniquement, et non depuis l' AWS API AWS CLI or.

Pour activer une clé d’accès ou une clé de sécurité pour l’utilisateur racine (console)

1. Ouvrez la console de gestion AWS et connectez-vous à l’aide de vos informations d’identification d’utilisateur racine.

   Pour obtenir des instructions, voir Se connecter en Console de gestion AWS tant qu'utilisateur root dans le guide de Connexion à AWS l'utilisateur.
2. À droite de la barre de navigation, choisissez le nom de votre compte, et cliquez sur Security credentials (Informations d'identification de sécurité).



3. Sur la page Mes identifiants de sécurité de votre utilisateur root, sous Multi-factor Authentification (MFA), choisissez Attribuer un appareil MFA.
4. Sur la page Nom du dispositif MFA, saisissez un Nom du dispositif, choisissez Clé d'accès ou Clé de sécurité, puis choisissez Suivant.
5. Dans Configurer le dispositif, configurez votre clé d'accès. Créez une clé d'accès avec des données biométriques telles que votre visage ou votre empreinte digitale, avec le code PIN d'un appareil, ou en insérant la clé de sécurité FIDO dans le port USB de votre ordinateur et en la touchant.
6. Suivez les instructions de votre navigateur pour choisir un fournisseur de clé d'accès ou l'endroit où vous souhaitez stocker votre clé d'accès afin de l'utiliser sur tous vos dispositifs.
7. Sélectionnez Continuer.

Vous avez maintenant enregistré votre clé d'accès pour l'utiliser avec AWS. La prochaine fois que vous utiliserez vos informations d'identification d'utilisateur racine pour vous connecter, vous devrez vous authentifier à l'aide de votre clé d'accès pour terminer le processus de connexion.

Pour obtenir de l'aide afin de résoudre les problèmes liés à votre clé de sécurité FIDO, consultez Résolution des problèmes liés aux clés d'accès et aux clés de sécurité FIDO.

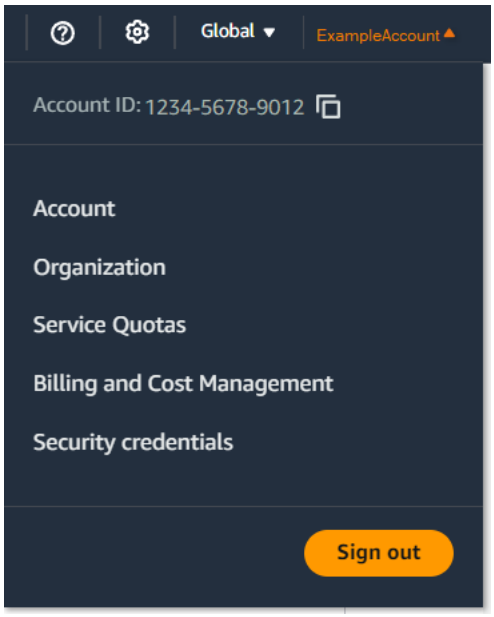
Activation d'un dispositif MFA virtuel pour l'utilisateur racine (console)

Vous pouvez utiliser le Console de gestion AWS pour configurer et activer un périphérique MFA virtuel pour votre utilisateur root. Pour activer les appareils MFA pour le Compte AWS, vous devez être connecté à AWS l'aide de vos informations d'identification d'utilisateur root.

Pour configurer et activer un dispositif MFA virtuel à utiliser avec votre utilisateur racine (console)

1. Ouvrez la console de gestion AWS et connectez-vous à l'aide de vos informations d'identification d'utilisateur racine.

   Pour obtenir des instructions, voir Se connecter en Console de gestion AWS tant qu'utilisateur root dans le Guide de Connexion à AWS l'utilisateur.
2. À droite de la barre de navigation, choisissez le nom de votre compte, et cliquez sur Security credentials (Informations d'identification de sécurité).



3. Dans la section Multi-Factor Authentification (MFA), choisissez Attribuer un appareil MFA.
4. Dans l'assistant, saisissez un nom dans le champ Nom du dispositif, sélectionnez Application Authenticator, puis cliquez sur Suivant.

   IAM génère et affiche les informations de configuration du dispositif MFA virtuel, notamment un graphique de code QR. Le graphique est une représentation de la clé de configuration secrète que l'on peut saisir manuellement sur des dispositifs qui ne prennent pas en charge les codes QR.
5. Ouvrez l'application MFA virtuelle sur l'appareil.

Si l'application MFA virtuelle prend en charge plusieurs comptes ou plusieurs dispositifs MFA virtuels, choisissez l'option permettant de créer un compte ou un dispositif MFA virtuel.

6. La manière la plus simple de configurer l'application consiste à utiliser l'application pour analyser le code QR. Si vous ne pouvez pas analyser le code, vous pouvez saisir les informations de configuration manuellement. Le code QR et la clé de configuration secrète générés par IAM sont liés à votre compte Compte AWS et ne peuvent pas être utilisés avec un autre compte. En revanche, ils peuvent être réutilisés pour configurer un nouveau dispositif MFA pour votre compte si vous perdez l'accès au dispositif MFA d'origine.

   - Pour utiliser le code QR pour configurer le dispositif MFA virtuel, dans l'assistant, choisissez Show QR code (Afficher le code QR). Ensuite, suivez les instructions de l'application pour scanner le code. Par exemple, vous pouvez avoir besoin de choisir l'icône de caméra ou une commande similaire à Scan account barcode (Analyser le code-barres du compte), puis d'utiliser la caméra du périphérique pour analyser le code QR.
   - Dans l'assistant Set up device (Configurer le dispositif), sélectionnez Show secret key (Afficher la clé secrète), puis saisissez la clé secrète dans votre application MFA.

   > **Important**
   >
   > Faites une sauvegarde sécurisée du code QR ou de la clé de configuration secrète ou assurez-vous d'activer plusieurs dispositifs MFA pour votre compte. Vous pouvez enregistrer jusqu'à huit appareils MFA de n'importe quelle combinaison des types MFA actuellement pris en charge auprès de vos utilisateurs Utilisateur racine d'un compte AWS et de ceux d'IAM. Un dispositif MFA virtuel peut devenir indisponible, par exemple, si vous perdez le smartphone hébergeant le dispositif MFA virtuel. Si cela se produit et que vous ne parvenez pas à vous connecter à votre compte sans autre dispositif MFA associé à l'utilisateur ou même d'après Récupération d'un dispositif MFA d'utilisateur racine, vous ne pourrez pas vous connecter à votre compte et vous devrez contacter le service client pour supprimer la protection MFA du compte.

   Le périphérique commence à générer des numéros à six chiffres.

7. Dans l'assistant, dans la zone MFA Code 1 (Code MFA 1), saisissez le mot de passe unique qui s'affiche actuellement sur le dispositif MFA virtuel. Attendez jusqu'à 30 secondes pour que le dispositif génère un nouveau mot de passe unique. Saisissez ensuite le second mot de passe

unique dans la zone MFA Code 2 (Code MFA 2). Choisissez Add MFA (Ajouter un dispositif MFA).

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré le code. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, le dispositif MFA s'associe avec succès à l'utilisateur mais est désynchronisé. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période. Dans ce cas, vous pouvez resynchroniser le dispositif.

L'appareil est prêt à être utilisé avec AWS. Pour plus d'informations sur l'utilisation de l'authentification MFA avec l'interface Console de gestion AWS, veuillez consulter Connexion compatible avec la MFA.

Activation d'un jeton TOTP matériel pour l'utilisateur root de l' (console)

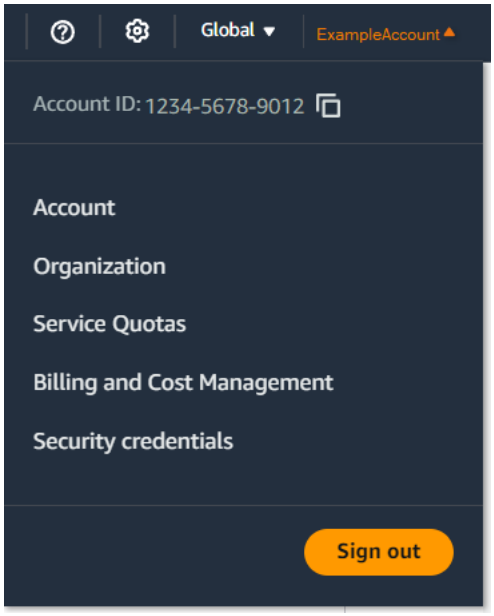
Vous pouvez configurer et activer un dispositif MFA physique pour votre utilisateur root Console de gestion AWS uniquement, et non à partir de l'API AWS CLI or AWS .

> **Note**
>
> Il se peut que vous remarquiez des textes différents, tels que se connecter à l'aide de MFA et dépanner votre dispositif d'authentification. Toutefois, les mêmes fonctions sont fournies. Dans les deux cas, si vous ne pouvez pas vérifier l'adresse e-mail et le numéro de téléphone de votre compte en utilisant d'autres facteurs d'authentification, contactez AWS Support pour supprimer votre configuration de MFA.

Pour activer un jeton TOTP matériel pour votre propre utilisateur racine (console)

1. Ouvrez la console de gestion AWS et connectez-vous à l'aide de vos informations d'identification d'utilisateur racine.

   Pour obtenir des instructions, voir Se connecter en Console de gestion AWS tant qu'utilisateur root dans le guide de Connexion à AWS l'utilisateur.
2. À droite de la barre de navigation, choisissez le nom de votre compte, et cliquez sur Security credentials (Informations d'identification de sécurité).



3. Développez la Multi-factor section Authentification (MFA).
4. Choisissez Assign MFA device (Attribuer un dispositif MFA).
5. Dans l'assistant, tapez le nom du dispositif, choisissez Hardware TOTP token (Jeton TOTP matériel), puis Next (Suivant).
6. Dans la zone Numéro de série, saisissez le numéro de série qui se trouve à l'arrière du dispositif MFA.
7. Dans la zone MFA code 1, saisissez le code à six chiffres qui s'affichent sur le dispositif MFA. Vous devrez peut-être appuyer sur le bouton situé à l'avant du périphérique pour afficher le numéro.



8. Attendez 30 secondes que le périphérique actualise le code, puis saisissez la nouvelle série de six chiffres dans la zone MFA code 2. Vous devrez peut-être appuyer à nouveau sur le bouton situé à l'avant du périphérique pour afficher le second numéro.
9. Choisissez Add MFA (Ajouter un dispositif MFA). Le dispositif MFA est à présent associé au Compte AWS.

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes d'authentification. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande,

l'dispositif MFA s'associe avec succès à l'utilisateur mais se désynchronise. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période. Dans ce cas, vous pouvez resynchroniser le dispositif.

La prochaine fois que vous utiliserez les informations d'identification d'utilisateur racine, vous devrez saisir un code du dispositif MFA.

Changez le mot de passe pour Utilisateur racine d'un compte AWS

Vous pouvez modifier l'adresse e-mail et le mot de passe à partir de la page Informations d'identification ou de la page Compte. Vous pouvez également sélectionner Mot de passe oublié ? sur la page de AWS connexion pour réinitialiser votre mot de passe.

Pour modifier le mot de passe de l'utilisateur root, vous devez vous connecter en tant qu'utilisateur IAM Utilisateur racine d'un compte AWS et non en tant qu'utilisateur. Pour savoir comment réinitialiser un mot de passe d'utilisateur racine oublié, veuillez consulter Réinitialisation d'un mot de passe d'utilisateur racine perdu ou oublié.

Pour protéger votre mot de passe, il est important de respecter les bonnes pratiques suivantes :

- Modifiez régulièrement votre mot de passe.
- Gardez votre mot de passe confidentiel, car toute personne qui le connaît peut accéder à votre compte.
- Utilisez un mot de passe différent AWS de celui que vous utilisez sur les autres sites.
- Evitez les mots de passe trop faciles à deviner, notamment : `secret`, `password`, `amazon` ou `123456`. Évitez également les mots du dictionnaire, votre nom, votre adresse électronique ou d'autres informations personnelles que quelqu'un pourrait facilement obtenir.

Important

Comptes AWS géré à l'aide d'un accès root centralisé AWS Organizations peut être activé pour les comptes membres. Ces comptes membres ne disposent pas d'informations d'identification de l'utilisateur racine, ne peuvent pas se connecter en tant qu'utilisateur racine et ne peuvent pas récupérer le mot de passe de l'utilisateur racine. Contactez votre

administrateur si vous devez effectuer une tâche qui nécessite les informations d'identification de l'utilisateur racine.

Console de gestion AWS

Pour changer le mot de passe de l'utilisateur racine

> **Autorisations minimales**
>
> Pour effectuer les étapes suivantes, vous devez au moins disposer des autorisations IAM suivantes :
>
> - Vous devez vous connecter en tant qu'utilisateur Compte AWS root, ce qui ne nécessite aucune autorisation supplémentaire Gestion des identités et des accès AWS (IAM). Vous ne pouvez pas effectuer ces étapes en tant qu'utilisateur ou rôle IAM.

1. Ouvrez la console de gestion AWS et connectez-vous à l'aide de vos informations d'identification d'utilisateur racine.

   Pour obtenir des instructions, voir Se connecter en Console de gestion AWS tant qu'utilisateur root dans le Guide de Connexion à AWS l'utilisateur.
2. Dans le coin supérieur droit de la console, choisissez votre nom ou votre numéro de compte, puis sélectionnez Informations d'identification de sécurité.
3. Sur la page Compte, en regard de Paramètres du compte, choisissez Modifier. Pour des raisons de sécurité, vous êtes invité à vous réauthentifier.

   > **Note**
   >
   > Si l'option Modifier ne s'affiche pas, il est probable que vous ne soyez pas connecté en tant qu'utilisateur root de votre compte. Vous ne pouvez pas modifier les paramètres du compte lorsque vous êtes connecté en tant qu'utilisateur ou rôle IAM.
4. Sur la page Mettre à jour les paramètres du compte, sous Mot de passe, sélectionnez Modifier.
5. Sur la page Mettre à jour votre mot de passe, remplissez les champs Mot de passe actuel, Nouveau mot de passe et Confirmer le nouveau mot de passe.

> **Important**
>
> Assurez-vous de choisir un mot de passe fort. Bien que vous puissiez définir une politique de mot de passe de compte pour les utilisateurs IAM, celle-ci ne s'applique pas à votre utilisateur root.

AWS exige que votre mot de passe remplisse les conditions suivantes :

- Avoir un minimum de 8 caractères et un maximum de 128 caractères
- Inclure au minimum trois des types de caractères suivants : majuscules, minuscules, chiffres, et les symboles ! @ # $ % ^ & * () <> [] {} | _ + - =
- Il ne doit pas être identique à votre Compte AWS nom ou à votre adresse e-mail.

6. Sélectionnez Enregistrer les modifications.

AWS CLI or AWS SDK

Cette tâche n'est pas prise en charge dans AWS CLI ou par une opération d'API depuis l'un des AWS kits SDK. Vous pouvez effectuer cette tâche uniquement à l'aide du Console de gestion AWS.

Réinitialisation d'un mot de passe d'utilisateur racine perdu ou oublié

Lorsque vous avez créé le vôtre pour la première fois Compte AWS, vous avez fourni une adresse e-mail et un mot de passe. Voici vos Utilisateur racine d'un compte AWS informations d'identification. Si vous oubliez votre mot de passe utilisateur racine, vous pouvez le réinitialiser à partir de la Console de gestion AWS.

Comptes AWS géré à l'aide d'un accès root centralisé AWS Organizations peut être activé pour les comptes membres. Ces comptes membres ne disposent pas d'informations d'identification de l'utilisateur racine, ne peuvent pas se connecter en tant qu'utilisateur racine et ne peuvent pas récupérer le mot de passe de l'utilisateur racine. Contactez votre administrateur si vous devez effectuer une tâche qui nécessite les informations d'identification de l'utilisateur racine.

> **Important**
>
> Vous rencontrez des difficultés pour vous connecter à AWS ? Assurez-vous que vous êtes sur la bonne page de connexion AWS pour votre type d'utilisateur. Si vous êtes le Utilisateur racine d'un compte AWS (propriétaire du compte), vous pouvez vous connecter à AWS l'aide des informations d'identification que vous avez configurées lors de la création du Compte AWS. Si vous êtes un utilisateur IAM, votre administrateur de compte peut vous fournir les informations d'identification que vous pouvez utiliser pour vous connecter à AWS. Si vous avez besoin d'assistance, n'utilisez pas le lien de commentaires sur cette page, car le formulaire n'est pas reçu par l'équipe de AWS documentation Support. Sur la page Contactez-nous, sélectionnez Impossible toujours de vous connecter à votre AWS compte, puis choisissez l'une des options d'assistance disponibles.

Pour réinitialiser votre mot de passe utilisateur racine

1. Ouvrez la console de gestion AWS et connectez-vous à l'aide de vos informations d'identification d'utilisateur racine.

   Pour obtenir des instructions, voir Se connecter en Console de gestion AWS tant qu'utilisateur root dans le Guide de Connexion à AWS l'utilisateur.

   > **Note**
   >
   > Si vous vous êtes connecté à Console de gestion AWS avec des informations d'identification d'utilisateur IAM, vous devez vous déconnecter pour pouvoir réinitialiser le mot de passe utilisateur racine. Si la page de connexion utilisateur IAM spécifique au compte s'affiche, choisissez d'Sign-in utiliser les informations d'identification du compte root en bas de la page. Si nécessaire, fournissez l'adresse e-mail de votre compte et choisissez Next (Suivant) pour accéder à la page Root user sign in (Connexion de l'utilisateur racine).

2. Choisissez Forgot your password? (Mot de passe oublié ?).

   > **Note**
   >
   > Si vous êtes un utilisateur IAM, cette option n'est pas disponible. L'option Mot de passe oublié ? n'est disponible que pour le compte utilisateur root. Les utilisateurs IAM doivent

demander à leur administrateur de réinitialiser un mot de passe oublié. Pour plus d'informations, consultez l'article J'ai oublié le mot de passe utilisateur IAM de mon AWS compte. Si vous vous connectez via le portail AWS d'accès, consultez Réinitialisation de votre mot de passe utilisateur IAM Identity Center.

3. Indiquez l'adresse e-mail associée au compte. Indiquez ensuite le texte CAPTCHA et choisissez Continue (Continuer).
4. Vérifiez que l'e-mail qui vous est associé ne contient Compte AWS pas de message provenant d'Amazon Web Services. L'e-mail peut provenir d'une adresse se terminant par dans `@verify.signin.aws`. Suivez les instructions de l'e-mail. Si vous ne voyez pas l'e-mail dans votre compte, vérifiez le dossier des courriers indésirables. Si vous n'avez plus accès à l'e-mail, consultez la section Je n'ai pas accès à l'e-mail correspondant à mon AWS compte dans le Guide de Connexion à AWS l'utilisateur.

Création de clés d'accès pour l'utilisateur racine

⚠ Warning

Nous vous recommandons vivement de ne pas créer de paires de clés d'accès pour votre utilisateur root. Étant donné que seules quelques tâches nécessitent l'utilisation de l'utilisateur root et que vous ne les effectuez généralement que rarement, nous vous recommandons de vous connecter au pour Console de gestion AWS effectuer les tâches de l'utilisateur root. Avant de créer des clés d'accès, passez en revue les alternatives aux clés d'accès à long terme.

Bien que cela ne soit pas recommandé, vous pouvez créer des clés d'accès pour votre utilisateur root afin de pouvoir exécuter des commandes dans le AWS Command Line Interface (AWS CLI) ou utiliser les opérations d'API depuis l'un des AWS kits SDK à l'aide des informations d'identification de l'utilisateur root. Lorsque vous créez des clés d'accès, vous créez l'ID de clé d'accès et la clé d'accès secrète sous forme d'ensemble. Lors de la création de la clé AWS d'accès, vous avez la possibilité de consulter et de télécharger la partie clé d'accès secrète de la clé d'accès. Si vous ne la téléchargez pas ou si vous la perdez, vous pouvez supprimez la clé d'accès, puis en créer une nouvelle. Vous pouvez créer des clés d'accès utilisateur root à l'aide de la console ou de AWS l'API. AWS CLI

Une clé d'accès nouvellement créée a le statut active, ce qui signifie que vous pouvez l'utiliser pour les appels de CLI et d'API. Vous pouvez attribuer jusqu'à deux clés d'accès à l'utilisateur root.

Les clés d'accès qui ne sont pas utilisées doivent être désactivées. Une fois qu'une clé d'accès est inactive, vous ne pouvez pas l'utiliser pour les appels d'API. Les clés inactives comptent toujours dans votre limite. Vous pouvez créer ou supprimer une clé d'accès à tout moment. Toutefois, la suppression d'une clé d'accès est définitive et vous ne pourrez plus la récupérer.

Console de gestion AWS

Pour créer une clé d'accès pour Utilisateur racine d'un compte AWS

> **Autorisations minimales**
>
> Pour effectuer les étapes suivantes, vous devez au moins disposer des autorisations IAM suivantes :
>
> - Vous devez vous connecter en tant qu'utilisateur Compte AWS root, ce qui ne nécessite aucune autorisation supplémentaire Gestion des identités et des accès AWS (IAM). Vous ne pouvez pas effectuer ces étapes en tant qu'utilisateur ou rôle IAM.

1. Ouvrez la console de gestion AWS et connectez-vous à l'aide de vos informations d'identification d'utilisateur racine.

   Pour obtenir des instructions, voir Se connecter en Console de gestion AWS tant qu'utilisateur root dans le Guide de Connexion à AWS l'utilisateur.
2. Dans le coin supérieur droit de la console, choisissez votre nom ou votre numéro de compte, puis sélectionnez Informations d'identification de sécurité.
3. Dans la section Clés d'accès, choisissez Créer une clé d'accès. Si cette option n'est pas disponible, cela signifie que vous avez déjà atteint le nombre maximum de clés d'accès. Vous devez supprimer l'une des clés d'accès existantes avant de pouvoir créer une clé. Pour plus d'informations, veuillez consulter Quotas d'objets IAM.
4. Sur la page Alternatives aux clés d'accès d'utilisateur root, passez en revue les recommandations de sécurité. Pour continuer, cochez la case, puis choisissez Créer une clé d'accès.
5. Sur la page Récupérer la clé d'accès, votre ID de clé d'accès est affiché.
6. Sous Clé d'accès secrète, choisissez Afficher puis copiez l'ID de la clé d'accès et la clé secrète dans la fenêtre de votre navigateur et collez-les dans un endroit sûr. Vous pouvez également choisir l'option Télécharger un fichier .csv qui téléchargera un fichier nommé

`rootkey.csv` contenant l'ID de la clé d'accès et la clé secrète. Enregistrez le fichier à un endroit sûr.

7. Sélectionnez Exécuté. Lorsque vous n'avez plus besoin d'utiliser la clé d'accès, nous vous recommandons de la supprimer, ou au moins d'envisager de la désactiver afin que personne ne puisse en abuser.

AWS CLI & SDKs

Pour créer une clé d'accès pour l'utilisateur root

> **Note**
>
> Pour exécuter la commande ou l'opération d'API suivante en tant qu'utilisateur root, vous devez déjà disposer d'une paire de clés d'accès active. Si vous n'avez aucune clé d'accès, créez la première clé d'accès à l'aide de l' Console de gestion AWS. Vous pouvez ensuite utiliser les informations d'identification de cette première clé d'accès AWS CLI pour créer la deuxième clé d'accès ou pour supprimer une clé d'accès.

- AWS CLI : aws iam create-access-key

  Example

```
$ aws iam create-access-key
{
    "AccessKey": {
        "UserName": "MyUserName",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "Status": "Active",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "CreateDate": "2021-04-08T19:30:16+00:00"
    }
}
```

- AWS API : CreateAccessKey dans la référence de l'API IAM.

Suppression d'une clé d'accès de l'utilisateur racine

Vous pouvez utiliser l' Console de gestion AWS API AWS CLI ou l' AWS API pour supprimer les clés d'accès de l'utilisateur root.

Console de gestion AWS

Pour supprimer une clé d'accès de l'utilisateur root

> **Autorisations minimales**
>
> Pour effectuer les étapes suivantes, vous devez au moins disposer des autorisations IAM suivantes :
>
> - Vous devez vous connecter en tant qu'utilisateur Compte AWS root, ce qui ne nécessite aucune autorisation Gestion des identités et des accès AWS (IAM) supplémentaire. Vous ne pouvez pas effectuer ces étapes en tant qu'utilisateur ou rôle IAM.

1. Ouvrez la console de gestion AWS et connectez-vous à l'aide de vos informations d'identification d'utilisateur racine.

   Pour obtenir des instructions, voir Se connecter en Console de gestion AWS tant qu'utilisateur root dans le guide de Connexion à AWS l'utilisateur.
2. Dans le coin supérieur droit de la console, choisissez votre nom ou votre numéro de compte, puis sélectionnez Informations d'identification de sécurité.
3. Dans la section Clés d'accès, sélectionnez la clé d'accès que vous souhaitez supprimer, puis sous Actions, choisissez Supprimer.

   > **Note**
   >
   > Vous pouvez également Désactiver une clé d'accès au lieu de la supprimer définitivement. Vous pourrez ainsi la réutiliser ultérieurement sans avoir à modifier l'ID de clé ou la clé secrète. Lorsque la clé est inactive, toute tentative de l'utiliser dans les demandes adressées à l' AWS API échoue et l'accès est refusé en cas d'erreur.
4. Dans la boîte de dialogue Supprimer <access key ID>, choisissez Désactiver, saisissez l'ID de la clé d'accès pour confirmer que vous souhaitez la supprimer, puis choisissez Supprimer.

AWS CLI & SDKs

Pour supprimer une clé d'accès de l'utilisateur root

> **Autorisations minimales**
>
> Pour effectuer les étapes suivantes, vous devez au moins disposer des autorisations IAM suivantes :
>
> - Vous devez vous connecter en tant qu'utilisateur Compte AWS root, ce qui ne nécessite aucune autorisation Gestion des identités et des accès AWS (IAM) supplémentaire. Vous ne pouvez pas effectuer ces étapes en tant qu'utilisateur ou rôle IAM.

- AWS CLI: cime delete-access-key

  Example

  ```
  $ aws iam delete-access-key \
      --access-key-id AKIAIOSFODNN7EXAMPLE
  ```

  Cette commande ne produit aucune sortie lorsqu'elle réussit.
- AWS API : DeleteAccessKey

Tâches nécessitant les informations d'identification de l'utilisateur root

Nous vous recommandons de configurer un utilisateur administratif AWS IAM Identity Center pour effectuer les tâches quotidiennes et accéder aux AWS ressources. Toutefois, vous ne pouvez effectuer les tâches énumérées ci-dessous que si vous vous connectez en tant qu'utilisateur root d'un compte.

Pour simplifier la gestion des informations d'identification des utilisateurs root privilégiés sur les comptes des membres AWS Organizations, vous pouvez activer l'accès root centralisé pour vous aider à sécuriser de manière centralisée un accès hautement privilégié à votre Comptes AWS. Gestion centralisée de l'accès root pour les comptes membresvous permet de supprimer de manière centralisée et d'empêcher la restauration à long terme des informations d'identification des utilisateurs root, améliorant ainsi la sécurité des comptes au sein de votre entreprise. Après avoir activé cette fonctionnalité, vous pouvez effectuer les tâches privilégiées suivantes sur les comptes membres.

- Supprimez les informations d'identification de l'utilisateur racine du compte membre pour empêcher la restauration du compte de l'utilisateur racine. Vous pouvez également autoriser la récupération du mot de passe pour récupérer les informations d'identification de l'utilisateur racine pour un compte membre.
- Supprimez une politique de compartiment mal configurée qui empêche tous les principaux d'accéder à un compartiment Amazon S3.
- Supprimez une politique Amazon Simple Queue Service qui refuse à tous les principaux l'accès à une file d'attente Amazon SQS.

Tâches de gestion de compte

- Modifiez vos paramètres Compte AWS . Les appareils autonomes Comptes AWS qui n'en font pas partie AWS Organizations nécessitent des informations d'identification root pour mettre à jour l'adresse e-mail, le mot de passe de l'utilisateur root et les clés d'accès de l'utilisateur root. Les autres paramètres du compte, tels que le nom du compte, les coordonnées, les contacts alternatifs, la devise de paiement préférée Régions AWS, ne nécessitent pas d'informations d'identification d'utilisateur root.

  > **Note**
  >
  > AWS Organizations, avec toutes les fonctionnalités activées, peut être utilisé pour gérer les paramètres des comptes membres de manière centralisée à partir du compte de gestion et des comptes d'administration délégués. Les utilisateurs IAM autorisés ou les rôles IAM dans le compte de gestion et dans les comptes d'administrateur délégué peuvent fermer les comptes des membres et mettre à jour les adresses e-mail racine, les noms des comptes, les informations de contact, les contacts secondaires et les comptes Régions AWS des membres.

- Fermez votre Compte AWS. Les appareils autonomes Comptes AWS qui n'en font pas partie AWS Organizations nécessitent des informations d'identification root pour fermer le compte. Avec AWS Organizations, vous pouvez fermer les comptes des membres de manière centralisée depuis le compte de gestion et les comptes d'administration délégués.
- Rouvrez votre. Compte AWS Si vous avez fermé vous-même un compte autonome et qu'il n'a aucun solde impayé, vous pouvez le rouvrir après sa fermeture. Pour ce faire, vous devez vous connecter en tant qu'utilisateur root.

- Restaurer les autorisations de l'utilisateur IAM. Si un utilisateur IAM révoque accidentellement ses propres autorisations, vous pouvez vous connecter en tant qu'utilisateur root pour modifier les politiques et restaurer ces autorisations.

Tâches de facturation

- Activer l'accès IAM à la console de gestion de la facturation et des coûts.
- Certaines tâches de facturation sont limitées à l'utilisateur racine. Consultez le guide de gestion Compte AWS d'un AWS Billing utilisateur intégré pour plus d'informations.
- Afficher certaines factures fiscales. Un utilisateur IAM disposant de l'ViewBillingautorisation aws-portal : peut consulter et télécharger les factures de TVA depuis AWS l'Europe, mais pas depuis AWS Inc. ou Amazon Internet Services Private Limited (AISPL).

AWS GovCloud (US) Tâches

- Inscrivez-vous à AWS GovCloud (US).
- Demandez les clés d'accès de l'utilisateur root au AWS GovCloud (US) compte auprès de AWS Support.

Tâches Amazon EC2

- Inscrit en tant que vendeur sur le Marketplace des instances réservées.

AWS KMS Sous-tâche

- Si une AWS Key Management Service clé devient ingérable, un administrateur peut la récupérer en contactant Support ; toutefois, il Support répond au numéro de téléphone principal de votre utilisateur root pour obtenir l'autorisation en confirmant le ticket OTP.

Tâches Amazon Simple Storage Service

- Configurer un compartiment Amazon S3 pour activer MFA (authentification multifactorielle).
- Modifier ou supprimer une politique de compartiment Amazon S3 qui refuse tous les principaux.

Vous pouvez utiliser des actions privilégiées pour déverrouiller un compartiment Amazon S3 dont la politique de compartiment est mal configurée. Pour plus de détails, consultez Effectuez une tâche privilégiée sur un AWS Organizations compte membre.

Tâche Amazon Simple Queue Service

- Modifier ou supprimer une politique de ressources Amazon SQS qui refuse tous les principaux.

  Vous pouvez utiliser des actions privilégiées pour déverrouiller une file d'attente Amazon SQS dont la politique basée sur les ressources est mal configurée. Pour plus de détails, consultez Effectuez une tâche privilégiée sur un AWS Organizations compte membre.

Ressources supplémentaires

Pour plus d'informations sur l'utilisateur AWS root, consultez les ressources suivantes :

- Pour obtenir de l'aide sur les problèmes liés aux utilisateurs racine, consultez Résolution de problèmes relatifs à l'utilisateur racine.
- Pour gérer de manière centralisée les adresses e-mail des utilisateurs root dans AWS Organizations, consultez la section Mise à jour de l'adresse e-mail de l'utilisateur root pour un compte membre dans le guide de AWS Organizations l'utilisateur.

Les articles suivants fournissent des informations supplémentaires sur l'utilisation de l'utilisateur root.

- Quelles sont les meilleures pratiques pour sécuriser mes ressources Compte AWS et les siennes ?
- Comment créer une règle d' EventBridge événement pour m'avertir que mon utilisateur root a été utilisé ?
- Surveiller et notifier l' Utilisateur racine d'un compte AWS activité
- Surveiller l'activité de l'utilisateur root IAM

Utilisateurs IAM

> **Important**
>
> Les meilleures pratiques IAM recommandent de demander aux utilisateurs humains d'utiliser la fédération avec un fournisseur d'identité pour accéder à l'aide d'informations d'identification temporaires au lieu d' AWS utiliser des utilisateurs IAM avec des informations d'identification à long terme. Nous vous recommandons de n'utiliser les utilisateurs IAM que pour les cas d'utilisation spécifiques non pris en charge par les utilisateurs fédérés.
> Si vous utilisez notre nouvelle AWS expérience et souhaitez permettre à des utilisateurs humains d'accéder à vos projets ou à vos AWS paramètres, vous devez les ajouter à votre équipe. Dans notre nouvelle expérience, vous ne pouvez pas créer d'utilisateurs IAM avec un accès à la console. Pour plus d'informations, voir Inviter des membres de l'équipe.

Un utilisateur IAM est une entité que vous créez dans votre Compte AWS. L'utilisateur IAM représente l'utilisateur humain ou la charge de travail qui utilise l'utilisateur IAM pour interagir avec les ressources AWS . Un utilisateur IAM se compose d'un nom et d'informations d'identification.

Un utilisateur IAM doté d'autorisations d'administrateur n'est pas un Utilisateur racine d'un compte AWS. Pour de plus amples informations sur l'utilisateur racine, veuillez consulter Utilisateur racine d'un compte AWS.

Comment ? AWS identifie un utilisateur IAM

Lorsque vous créez un utilisateur IAM, IAM implémente les éléments suivants pour l'identifier :

- Un « nom convivial » pour l'utilisateur IAM. Il s'agit du nom que vous avez spécifié lors de la création de l'utilisateur IAM, par exemple `Richard` ou `Anaya`. Ce sont les noms que vous voyez dans Console de gestion AWS. Étant donné que les noms d'utilisateur IAM apparaissent dans les noms de ressources Amazon (ARN), nous vous recommandons de ne pas inclure d'informations d'identification personnelle dans le nom IAM. Reportez-vous à Exigences relatives aux noms IAM pour connaître les exigences et les restrictions relatives aux noms IAM.
- Un Amazon Resource Name (ARN) pour l'utilisateur IAM. Vous utilisez l'ARN lorsque vous devez identifier de manière unique l'utilisateur IAM sur tous les sites. AWS Par exemple, vous pouvez utiliser un ARN pour spécifier l'utilisateur IAM en tant que `Principal` dans la politique IAM d'un compartiment Amazon S3. L'ARN d'un utilisateur IAM peut se présenter comme suit :

`arn:aws:iam::account-ID-without-hyphens:user/Richard`

- Un identifiant unique pour l'utilisateur IAM. Cet ID est renvoyé uniquement lorsque vous utilisez l'API, Tools for Windows PowerShell ou AWS CLI pour créer l'utilisateur IAM ; il n'apparaît pas dans la console.

Pour plus d'informations sur ces identifiants, consultez Identifiants IAM.

Utilisateurs IAM et informations d'identification

Vous pouvez y accéder de différentes manières AWS en fonction des informations d'identification de l'utilisateur IAM :

- Mot de passe de la console : un mot de passe que l'utilisateur IAM peut entrer pour se connecter à des sessions interactives telles que l' Console de gestion AWS. La désactivation du mot de passe (accès à la console) d'un utilisateur IAM l'empêche de s'y connecter à l' Console de gestion AWS aide de ses informations de connexion. Cela ne modifie pas ses autorisations ni ne l'empêche d'accéder à la console à l'aide d'un rôle endossé. Les utilisateurs IAM dont l'accès à la console est activé peuvent également utiliser ces mêmes informations d'identification pour s'authentifier AWS CLI et accéder au SDK à l'aide de la commande. `aws login` AWS CLI Ces utilisateurs devront disposer d'SignInLocalDevelopmentAccessautorisations. Consultez la section Authentification et informations d'accès AWS CLI dans le Guide de AWS Command Line Interface l'utilisateur pour plus de détails.
- Clés d'accès : utilisées pour passer des appels programmatiques à AWS. Toutefois, il existe des alternatives plus sécurisées à envisager avant de créer des clés d'accès pour les utilisateurs IAM. Pour plus d'informations, consultez Considérations et alternatives relatives aux clés d'accès à long terme dans Références générales AWS. Si l'utilisateur IAM possède des clés d'accès actives, celles-ci continuent de fonctionner et autorisent l'accès via les AWS CLI outils pour Windows PowerShell, AWS l'API ou l'application mobile de la AWS console.
- Clés SSH à utiliser avec CodeCommit : une clé publique SSH au format OpenSSH qui peut être utilisée pour s'authentifier auprès d' CodeCommit.
- Certificats de serveur : SSL/TLS certificats que vous pouvez utiliser pour vous authentifier auprès de certains AWS services. Nous vous recommandons d'utiliser AWS Certificate Manager (ACM) pour provisionner, gérer et déployer vos certificats de serveur. Utilisez IAM uniquement lorsque vous devez prendre en charge des connexions HTTPS dans une région non prise en charge par ACM. Pour savoir quelles régions prennent en charge ACM, consultez Points de terminaison et quotas AWS Certificate Manager dans Références générales AWS.

Vous pouvez choisir les informations d'identification qui conviennent à votre utilisateur IAM. Lorsque vous utilisez la Console de gestion AWS pour créer un utilisateur IAM, vous devez au moins choisir d'inclure un mot de passe ou des clés d'accès à la console. Par défaut, un tout nouvel utilisateur IAM créé à l'aide de l' AWS API AWS CLI or ne possède aucune information d'identification. Vous devez créer le type d'informations d'identification pour un utilisateur IAM en fonction de votre cas d'utilisation.

Vous disposez des options suivantes pour administrer les mots de passe, les clés d'accès et les dispositifs d'authentification multifactorielle (MFA) :

- Gérer les mots de passe pour vos utilisateurs IAM. Créez et modifiez les mots de passe permettant d'accéder à Console de gestion AWS. Définissez une politique de mot de passe afin d'appliquer une complexité minimale pour les mots de passe. Autorisez les utilisateurs IAM à modifier leurs propres mots de passe.
- Gérer les clés d'accès pour vos utilisateurs IAM. Créez et mettez à jour les clés d'accès afin de permettre l'accès par programmation aux ressources de votre compte.
- Activez l'authentification multifactorielle (MFA) pour l'utilisateur IAM. Selon les bonnes pratiques, nous vous recommandons d'exiger une authentification multifactorielle pour tous les utilisateurs IAM de votre compte. Avec MFA, les utilisateurs IAM doivent fournir deux formes d'identification : tout d'abord, ils fournissent les informations d'identification faisant partie de leur identité utilisateur (un mot de passe ou une clé d'accès). En outre, ils fournissent un code numérique temporaire généré sur un matériel ou par une application sur un smartphone ou une tablette.
- Recherche de mots de passe et clés d'accès inutilisés. Toute personne possédant un mot de passe ou des clés d'accès pour votre compte ou un utilisateur IAM associé à votre compte a accès à vos AWS ressources. En matière de sécurité, les pratiques exemplaires consistent à supprimer les mots de passe et les clés d'accès dont les utilisateurs IAM n'ont plus besoin.
- Téléchargez un rapport d'informations d'identification pour votre compte. Vous pouvez générer et télécharger un rapport sur les informations d'identification qui répertorie tous les utilisateurs IAM de votre compte et le statut de leurs diverses informations d'identification, notamment leurs mots de passe, clés d'accès et dispositifs MFA. Pour les mots de passe et les clés d'accès, le rapport d'informations d'identification indique leur dernière date d'utilisation.

Rôles et autorisations IAM

Par défaut, un nouvel utilisateur IAM ne dispose d'aucune autorisation pour faire quoi que ce soit. Ils ne sont pas autorisés à effectuer des AWS opérations ou à accéder à des AWS ressources. Lorsque

vous configurez des utilisateurs IAM individuels, il est également possible de leur affecter des autorisations individuellement. Vous pouvez attribuer des autorisations administratives à quelques utilisateurs, qui peuvent ensuite administrer vos AWS ressources et même créer et gérer d'autres utilisateurs IAM. Dans la plupart des cas, toutefois, vous souhaitez limiter les autorisations d'un utilisateur aux seules tâches (AWS actions ou opérations) et aux ressources nécessaires à la tâche.

Prenons l'exemple d'un utilisateur nommé Diego. Lorsque vous créez l'utilisateur IAM `Diego`, vous lui créez un mot de passe et lui attachez des autorisations qui lui permettent de lancer une instance Amazon EC2 spécifique et de lire (`GET`) les informations d'une table dans une base de données Amazon RDS. Pour les procédures relatives à la création des utilisateurs IAM et à l'octroi d'informations d'identification initiales, consultez Créez un utilisateur IAM dans votre Compte AWS. Pour les procédures relatives à la modification des autorisations pour les utilisateurs existants, consultez Modification des autorisations pour un utilisateur IAM. Pour les procédures relatives à la modification des clés d'accès et du mot de passe de l'utilisateur, consultez Mots de passe utilisateur dans AWS et Gestion des clés d'accès pour les utilisateurs IAM.

Vous pouvez également ajouter une limite des autorisations à vos utilisateurs IAM. Une limite d'autorisations est une fonctionnalité avancée qui vous permet d'utiliser des politiques AWS gérées pour limiter les autorisations maximales qu'une politique basée sur l'identité peut accorder à un utilisateur ou à un rôle IAM. Pour plus d'informations sur les types de politiques et les utilisations, veuillez consulter Politiques et autorisations dans Gestion des identités et des accès AWS.

Utilisateurs et comptes IAM

Chaque utilisateur IAM est associé à un seul et même Compte AWS. Comme les utilisateurs IAM sont définis au sein de votre organisation Compte AWS, il n'est pas nécessaire qu'ils disposent d'un mode de paiement enregistré auprès AWS de. Toute AWS activité effectuée par les utilisateurs IAM sur votre compte est facturée sur votre compte.

Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour plus d'informations, consultez IAM et AWS STS quotas.

Utilisateurs IAM en tant que comptes de service

Un utilisateur IAM est une ressource dans IAM à laquelle des informations d'identification et ses autorisations sont associées. Un utilisateur IAM peut représenter une personne ou une application qui utilise ses informations d'identification pour exécuter des demandes AWS . En général, ceci est appelé un compte de service. Si vous choisissez d'utiliser les informations d'identification à long

terme d'un utilisateur IAM dans votre application, n'incorporez pas directement les clés d'accès dans le code de votre application. Les AWS SDK et les autres vous AWS Command Line Interface permettent de placer les clés d'accès dans des emplacements connus afin de ne pas avoir à les conserver dans le code. Pour de plus amples informations, consultez Gestion correcte des clés d'accès utilisateur IAM (français non garanti) dans Références générales AWS. En tant que bonne pratique, vous pouvez également utiliser des informations d'identification de sécurité temporaires (rôles IAM) au lieu des clés d'accès à long terme.

Une fois que vous avez créé des utilisateurs IAM, vous pouvez Affichage des utilisateurs IAM accéder à votre compte et Renommer un utilisateur IAM selon vos besoins.

Comment les utilisateurs d'IAM se connectent à AWS

Pour vous connecter en Console de gestion AWS tant qu'utilisateur IAM, vous devez fournir votre identifiant de compte ou votre alias de compte en plus de votre nom d'utilisateur et de votre mot de passe. Lorsque votre administrateur a créé votre utilisateur IAM dans la console, il a dû vous envoyer vos informations d'identification de connexion, notamment votre nom d'utilisateur et l'URL de la page de connexion à votre compte, qui inclut votre ID de compte ou votre alias de compte.

```
https://My_AWS_Account_ID.signin.aws.amazon.com/console/
```

> **Conseil**
>
> Pour créer un marque-page pour la page de connexion à votre compte dans votre navigateur web, vous devez saisir manuellement l'URL de connexion de votre compte lors de la création du marque-page. N'utilisez pas la fonction « Marquer cette page » de votre navigateur web, en raison des redirections qui risquent de masquer l'URL de connexion.

Vous pouvez également vous connecter au point de terminaison général suivant et saisir manuellement votre ID de compte ou votre alias de compte :

```
https://console.aws.amazon.com/
```

Pour plus de commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser le nom d'utilisateur IAM et les informations de compte. La prochaine fois que l'utilisateur accède à une page du Console de gestion AWS, la console utilise le cookie pour le rediriger vers la page de connexion au compte.

Vous n'avez accès qu'aux AWS ressources spécifiées par votre administrateur dans la politique associée à votre identité d'utilisateur IAM. Pour travailler dans la console, vous devez disposer des autorisations nécessaires pour effectuer les actions effectuées par la console, telles que la liste et la création de AWS ressources. Pour plus d'informations, consultez Gestion des accès pour AWS resources et Exemples de politiques basées sur l'identité IAM.

> **Note**
>
> Si votre organisation dispose déjà d'un système d'identité, vous souhaiterez peut-être créer une option d'authentification unique (SSO). Le SSO permet aux utilisateurs d'accéder Console de gestion AWS à votre compte sans qu'ils aient à disposer d'une identité d'utilisateur IAM. Le SSO élimine également la nécessité pour les utilisateurs de se connecter au site de votre organisation et de le faire AWS séparément. Pour de plus amples informations, veuillez consulter Activez l'accès personnalisé du courtier d'identité au AWS console.

Enregistrement des informations de connexion dans CloudTrail

Si vous activez CloudTrail l'enregistrement des événements de connexion dans vos journaux, vous devez savoir comment CloudTrail choisir où enregistrer les événements.

- Si les utilisateurs se connectent directement à une console, ils sont redirigés vers un point de terminaison de connexion international ou régional, si la console de service sélectionnée prend en charge les régions. Par exemple, la page d'accueil de la console principale prend en charge les régions, donc si vous vous connectez à l'URL suivante :

  ```
  https://alias.signin.aws.amazon.com/console
  ```

  vous êtes redirigé vers un point de terminaison de connexion régional tel que`https://us-east-2.signin.aws.amazon.com`, ce qui entraîne une entrée de CloudTrail journal régional dans le journal de la région de l'utilisateur :

  En revanche, la console Amazon S3 ne prend pas en charge les régions, donc si vous vous connectez à l'URL suivante

  ```
  https://alias.signin.aws.amazon.com/console/s3
  ```

AWS vous redirige vers le point de terminaison de connexion global à`https://signin.aws.amazon.com`, ce qui entraîne une entrée de CloudTrail journal globale.

- Vous pouvez demander manuellement un point de terminaison de connexion régional spécifique en vous connectant à la page d'accueil régionale de la console principale à l'aide d'une syntaxe d'URL similaire à ce qui suit :

```
https://alias.signin.aws.amazon.com/console?region=ap-southeast-1
```

AWS vous redirige vers le point de terminaison de connexion `ap-southeast-1` régional et entraîne un événement de CloudTrail journalisation régional.

Pour plus d'informations sur l'IAM CloudTrail et l'IAM, consultez la section Journalisation des événements IAM avec. CloudTrail

Si les utilisateurs ont besoin d'un accès par programmation pour utiliser votre compte, vous pouvez créer une paire de clés d'accès (un ID de clé d'accès et une clé d'accès secrète) pour chaque utilisateur. Toutefois, il existe des alternatives plus sécurisées à envisager avant de créer des clés d'accès pour les utilisateurs. Pour plus d'informations, consultez Considérations et alternatives relatives aux clés d'accès à long terme dans Références générales AWS.

Ressources supplémentaires

Les ressources suivantes peuvent vous aider à en savoir plus sur la AWS connexion.

- Le guide de AWS Sign-In l'utilisateur vous aide à comprendre les différentes manières de vous connecter à Amazon Web Services (AWS), en fonction de votre type d'utilisateur.
- Vous pouvez vous connecter à un maximum de cinq identités différentes simultanément dans un seul navigateur Web dans l' Console de gestion AWS. Pour plus de détails, consultez la section Connexion à plusieurs comptes dans le Guide de démarrage Console de gestion AWS .
- Pour trouver l'ID de compte ou l'alias requis pour la connexion, consultez Visualisation de votre Compte AWS ID etUtiliser un alias pour votre Compte AWS ID.

Connexion compatible avec la MFA

Les utilisateurs pour lesquels l'authentification multifactorielle (MFA) est configurée doivent utiliser leurs dispositifs MFA pour se connecter à l' Console de gestion AWS. Une fois que l'utilisateur a saisi ses informations de connexion, AWS vérifie dans son compte si le MFA est requis pour cet utilisateur.

> **Important**
>
> Si vous utilisez des informations d'identification par clé d'accès et clé secrète pour Console de gestion AWS un accès direct AWS STS `GetFederationToken`via l'appel d'API, la MFA ne sera PAS requise. Pour de plus amples informations, veuillez consulter Utilisation des clés d'accès et des informations d'identification de clé secrète pour l'accès à la console.

Les rubriques suivantes fournissent des informations sur la façon dont les utilisateurs effectuent leur connexion lorsque l'authentification MFA est requise.

Rubriques

- Plusieurs dispositifs MFA activés
- Clé de sécurité FIDO
- Dispositif MFA virtuel
- Jeton TOTP matériel

Plusieurs dispositifs MFA activés

Si un utilisateur se connecte en Console de gestion AWS tant qu'utilisateur Compte AWS root ou utilisateur IAM avec plusieurs appareils MFA activés pour ce compte, il n'a besoin d'utiliser qu'un seul appareil MFA pour se connecter. Une fois que l'utilisateur s'est authentifié à l'aide du mot de passe de l'utilisateur, il sélectionne le type de dispositif MFA qu'il souhaite utiliser pour terminer l'authentification. L'utilisateur est ensuite invité à s'authentifier avec le type de dispositif qu'il a sélectionné.

Clé de sécurité FIDO

Si l'authentification MFA est requise pour l'utilisateur, une deuxième page de connexion s'affiche. L'utilisateur a besoin d'utiliser la clé de sécurité FIDO.

> **Note**
>
> Les utilisateurs de Google Chrome ne doivent choisir aucune des options disponibles dans la fenêtre contextuelle qui demande Verify your identity with amazon.com (Vérifiez votre identité auprès d'amazon.com). Il vous suffit d'appuyer sur la clé de sécurité.

Contrairement aux autres dispositifs MFA, les clés de sécurité FIDO ne se désynchronisent pas. Les administrateurs peuvent désactiver une clé de sécurité FIDO si elle est perdue ou endommagée. Pour de plus amples informations, veuillez consulter Désactivation des dispositifs MFA (console).

Pour plus d'informations sur les navigateurs compatibles WebAuthn et les appareils compatibles FIDO compatibles, AWS consultez. Configurations prises en charge pour l'utilisation des clés d'accès et des clés de sécurité

Dispositif MFA virtuel

Si l'authentification MFA est requise pour l'utilisateur, une deuxième page de connexion s'affiche. Dans le champ Code MFA, l'utilisateur doit entrer le code numérique fourni par l'application MFA.

Si le code MFA est correct, l'utilisateur peut accéder à l'interface Console de gestion AWS. Si le code est incorrect, l'utilisateur peut refaire une tentative avec un autre code.

Un dispositif MFA virtuel peut être désynchronisé. Si un utilisateur ne parvient pas à se connecter Console de gestion AWS après plusieurs tentatives, il est invité à synchroniser le périphérique MFA virtuel. L'utilisateur peut suivre les instructions affichées à l'écran pour synchroniser le dispositif MFA virtuel. Pour plus d'informations sur la façon dont vous pouvez synchroniser un appareil pour le compte d'un utilisateur de votre Compte AWS, consultezRésynchronisation des dispositifs MFA virtuels et matériels.

Jeton TOTP matériel

Si l'authentification MFA est requise pour l'utilisateur, une deuxième page de connexion s'affiche. Dans le champ Code MFA, l'utilisateur doit entrer le code numérique fourni par un jeton TOTP matériel.

Si le code MFA est correct, l'utilisateur peut accéder à l'interface Console de gestion AWS. Si le code est incorrect, l'utilisateur peut refaire une tentative avec un autre code.

Un jeton TOTP matériel peut être désynchronisé. Si un utilisateur ne parvient pas à se connecter Console de gestion AWS après plusieurs tentatives, il est invité à synchroniser le dispositif à jetons

MFA. L'utilisateur peut suivre les instructions affichées à l'écran pour synchroniser le dispositif de jeton MFA. Pour plus d'informations sur la façon dont vous pouvez synchroniser un appareil pour le compte d'un utilisateur de votre Compte AWS, consultezResynchronisation des dispositifs MFA virtuels et matériels.

Créez un utilisateur IAM dans votre Compte AWS

> **Important**
>
> Les meilleures pratiques IAM recommandent de demander aux utilisateurs humains d'utiliser la fédération avec un fournisseur d'identité pour accéder à l'aide d'informations d'identification temporaires au lieu d' AWS utiliser des utilisateurs IAM avec des informations d'identification à long terme. Nous vous recommandons de n'utiliser les utilisateurs IAM que pour les cas d'utilisation spécifiques non pris en charge par les utilisateurs fédérés.
> Si vous utilisez notre nouvelle AWS expérience et souhaitez permettre à des utilisateurs humains d'accéder à vos projets ou à vos AWS paramètres, vous devez les ajouter à votre équipe. Dans notre nouvelle expérience, vous ne pouvez pas créer d'utilisateurs IAM avec un accès à la console. Pour plus d'informations, voir Inviter des membres de l'équipe.

Le processus de création d'un utilisateur IAM et sa capacité à exécuter des tâches se compose des étapes suivantes :

1. Créez l'utilisateur dans Console de gestion AWS les AWS CLI Outils pour Windows PowerShell ou à l'aide d'une opération d' AWS API. Si vous créez l'utilisateur dans le Console de gestion AWS, les étapes 1 à 4 sont gérées automatiquement, en fonction de vos choix. Si vous créez les utilisateurs IAM par programme, vous devez exécuter chacune de ces étapes individuellement.
2. Créez les informations d'identification pour l'utilisateur, en fonction du type d'accès dont il a besoin :
   - Activer l'accès à la console — facultatif : si l'utilisateur doit accéder au Console de gestion AWS, créez un mot de passe pour l'utilisateur. La désactivation de l'accès à la console pour un utilisateur l'empêche de se connecter à l' Console de gestion AWS à l'aide de son nom d'utilisateur et de son mot de passe. Cela ne modifie pas ses autorisations ni ne l'empêche d'accéder à la console à l'aide d'un rôle endossé.

> **Tip**
>
> Créez uniquement les informations d'identification dont a besoin l'utilisateur. Par exemple, pour un utilisateur nécessitant un accès uniquement via le Console de gestion AWS, ne créez pas de clés d'accès.

3. Donner à l'utilisateur les autorisations nécessaires pour effectuer les tâches requises. Nous vous recommandons de placer vos utilisateurs IAM dans des groupes et de gérer leurs autorisations par le biais de politiques attachées à ces groupes. Toutefois, vous pouvez également accorder des autorisations en attachant des politiques d'autorisations directement à l'utilisateur. Si vous utilisez la console pour ajouter l'utilisateur, vous pouvez copier les autorisations d'un utilisateur existant vers le nouvel utilisateur.

   Vous pouvez également ajouter une limite des autorisations pour limiter les autorisations de l'utilisateur en spécifiant une politique qui définit les autorisations maximales que l'utilisateur peut avoir. Les limites des autorisations n'accordent aucune autorisation.

   Pour obtenir des instructions sur la création d'une politique d'autorisation personnalisée à utiliser pour accorder des autorisations ou définir une limite des autorisations, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client.
4. (Facultatif) Ajoutez des métadonnées à l'utilisateur en associant des balises. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Tags pour Gestion des identités et des accès AWS resources.
5. Fournissez à l'utilisateur les informations nécessaires à la connexion. Cela inclut le mot de passe et l'URL de la console de la page de connexion au compte sur laquelle l'utilisateur saisit ces informations d'identification. Pour plus d'informations, veuillez consulter Comment les utilisateurs d'IAM se connectent à AWS.
6. (Facultatif) Configurez l'authentification multifacteur (MFA) pour l'utilisateur. MFA exige que l'utilisateur fournisse un code à utilisation unique chaque fois qu'il se connecte à Console de gestion AWS.
7. (Facultatif) Accordez aux utilisateurs IAM les autorisations requises pour gérer leurs propres informations d'identification. (Par défaut, les utilisateurs IAM ne sont pas autorisés à gérer leurs propres informations d'identification.) Pour plus d'informations, consultez Octroi aux utilisateurs IAM de l'autorisation de modification de leurs propres mots de passe.

> **Note**
>
> Si vous utilisez la console pour créer l'utilisateur et que vous sélectionnez l'Utilisateur doit créer un nouveau mot de passe à la prochaine connexion (recommandé), l'utilisateur dispose des autorisations requises.

Pour plus d'informations sur les autorisations requises pour créer un utilisateur, consultez Autorisations requises pour accéder aux autres ressources IAM.

Pour de plus amples informations sur la création d'utilisateurs IAM pour des cas d'utilisation particuliers, consultez les rubriques suivantes :

- Création d'un utilisateur IAM pour l'accès d'urgence
- Création d'un utilisateur IAM pour les charges de travail qui ne peuvent pas utiliser de rôles IAM

Affichage des utilisateurs IAM

Vous pouvez répertorier les utilisateurs IAM de votre groupe IAM Compte AWS ou d'un groupe IAM spécifique, et répertorier tous les groupes IAM auxquels appartient un utilisateur. Pour plus d'informations sur les autorisations requises pour répertorier des utilisateurs, consultez Autorisations requises pour accéder aux autres ressources IAM.

Pour répertorier tous les utilisateurs IAM de votre compte

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d'accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez utilisateurs.

La console affiche les utilisateurs IAM de votre Compte AWS.

AWS CLI

Exécutez la commande suivante :

- `aws iam list-users`

API

Appelez l'opération suivante :

- `ListUsers`

Pour répertorier les utilisateurs dans un groupe IAM

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d'accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs).
4. Choisissez le nom du groupe d'utilisateurs.

Les utilisateurs IAM membres du groupe sont répertoriés dans l'onglet Utilisateurs.

AWS CLI

Exécutez la commande suivante :

- `aws iam get-group`

API

Appelez l'opération suivante :

- `GetGroup`

Pour répertorier tous les groupes IAM auxquels appartient un utilisateur

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d'accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez utilisateurs.
4. Dans la liste des Users (Utilisateurs), choisissez le nom d'utilisateur IAM.
5. Sélectionnez l'onglet Groupes pour afficher la liste des groupes qui incluent l'utilisateur actuel.

AWS CLI

Exécutez la commande suivante :

- `aws iam list-groups-for-user`

API

Appelez l'opération suivante :

- `ListGroupsForUser`

Étapes suivantes

Une fois que vous avez une liste de vos utilisateurs IAM, vous pouvez renommer, supprimer ou désactiver un utilisateur IAM en suivant les procédures suivantes.

- Renommer un utilisateur IAM
- Suppression ou désactivation d'un utilisateur IAM

Renommer un utilisateur IAM

> **Note**
>
> En tant que bonne pratique, nous vous recommandons de demander aux utilisateurs humains d'utiliser la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires. Si vous suivez les bonnes pratiques, vous ne gérez pas les utilisateurs et les groupes IAM. Au lieu de cela, vos utilisateurs et vos groupes sont gérés en dehors de l'extérieur AWS et peuvent accéder aux AWS ressources en tant qu'identité fédérée. Une identité fédérée est un utilisateur de l'annuaire des utilisateurs de votre entreprise, un fournisseur d'identité Web, le AWS Directory Service, le répertoire Identity Center ou tout utilisateur qui accède AWS aux services à l'aide des informations d'identification fournies par le biais d'une source d'identité. Les identités fédérées utilisent les groupes définis par leur fournisseur d'identité. Si vous en utilisez AWS IAM Identity Center, consultez la section Gérer les identités dans IAM Identity Center dans le Guide de l'AWS IAM Identity Center utilisateur pour plus d'informations sur la création d'utilisateurs et de groupes dans IAM Identity Center.

Amazon Web Services fournit plusieurs outils permettant de gérer les utilisateurs IAM dans votre Compte AWS. Vous pouvez répertorier les utilisateurs IAM de votre compte ou d'un groupe d'utilisateurs, ou dresser la liste de tous les groupes IAM auxquels appartient un utilisateur. Vous pouvez renommer ou modifier le chemin d'accès d'un utilisateur IAM. Si vous passez à l'utilisation d'identités fédérées au lieu d'utilisateurs IAM, vous pouvez supprimer un utilisateur IAM de votre compte AWS ou désactiver l'utilisateur.

Pour plus d'informations sur l'ajout, la modification ou la suppression de politiques gérées pour un utilisateur IAM, consultez Modification des autorisations pour un utilisateur IAM. Pour plus d'informations sur la gestion des politiques en ligne pour les utilisateurs IAM, consultez Ajout et suppression d'autorisations basées sur l'identité IAM, Modification de politiques IAM et Suppression de politiques IAM. En guise de bonne pratique, utilisez des politiques gérées plutôt que des politiques en ligne. Les politiques gérées par AWS octroient des autorisations pour de nombreux cas d'utilisation courants. N'oubliez pas que les politiques AWS gérées peuvent ne pas accorder d'autorisations de moindre privilège pour vos cas d'utilisation spécifiques, car elles peuvent être utilisées par tous les AWS clients. En conséquence, nous vous recommandons de réduire encore les autorisations en définissant des Politiques gérées par le client qui sont spécifiques à vos cas d'utilisation. Pour de plus amples informations, veuillez consulter AWS stratégies gérées. Pour

plus d'informations sur les politiques AWS gérées conçues pour des fonctions professionnelles spécifiques, consultezAWS politiques gérées pour les fonctions professionnelles.

Pour en savoir plus sur la validation des politiques IAM, consultez Validation de politique IAM.

> **Tip**
>
> IAM Access Analyzer peut analyser les services et les actions que vos rôles IAM utilisent, puis générer une politique précise que vous pouvez utiliser. Après avoir testé chaque stratégie générée, vous pouvez la déployer dans votre environnement de production. Cela garantit que vous n'accordez que les autorisations requises à vos charges de travail. Pour plus d'informations sur la génération de politiques, consultez IAM Access Analyzer policy generation.

Pour de plus amples informations sur la gestion des mots de passe utilisateur IAM, consultez Gérer les mots de passe des utilisateurs IAM.

Renommer un utilisateur IAM

Pour modifier le nom ou le chemin d'un utilisateur AWS CLI, vous devez utiliser l' AWS API Outils pour Windows PowerShell. La console ne comporte aucune option permettant de renommer un utilisateur. Pour plus d'informations sur les autorisations requises pour renommer un utilisateur, consultez Autorisations requises pour accéder aux autres ressources IAM.

Lorsque vous modifiez le nom ou le chemin d'accès d'un utilisateur, voici ce qui suit se produit :

- Toutes les politiques attachées à l'utilisateur restent dans l'utilisateur sous le nouveau nom.
- L'utilisateur reste dans les mêmes groupes IAM sous le nouveau nom.
- L'ID unique de l'utilisateur demeure le même. Pour plus d'informations sur le caractère unique IDs, consultezIdentifiants uniques.
- Toutes les ressources ou politiques de rôle qui font référence à l'utilisateur en tant que principal (l'utilisateur se voit accorder l'accès) sont mises à jour automatiquement pour utiliser le nouveau nom ou chemin. Par exemple, toutes les politiques basées sur des files d'attente dans Amazon SQS ou les politiques basées sur des ressources dans Amazon S3 sont mises à jour automatiquement pour utiliser le nouveau nom ou chemin.

IAM ne met pas automatiquement à jour les politiques qui font référence à l'utilisateur en tant que ressource de manière à utiliser le nouveau nom ou chemin ; vous devez le faire manuellement. Par exemple, imaginons qu'une politique est attachée à l'utilisateur `Richard` et qu'elle lui permet de gérer ses informations d'identification de sécurité. Si un administrateur renomme `Richard` en `Rich`, administrateur doit également mettre à jour cette politique pour changer la ressource de :

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Richard
```

à :

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Rich
```

C'est également vrai si un administrateur modifie le chemin : il doit également mettre à jour la politique pour refléter le nouveau chemin de l'utilisateur.

Pour renommer un utilisateur

- AWS CLI : aws iam update-user
- AWS API : UpdateUser

Suppression ou désactivation d'un utilisateur IAM

Les meilleures pratiques recommandent de supprimer les utilisateurs IAM non utilisés de votre Compte AWS. Si vous souhaitez conserver les informations d'identification de l'utilisateur IAM pour une utilisation ultérieure, au lieu de les supprimer du compte, vous pouvez désactiver l'accès de l'utilisateur. Pour de plus amples informations, veuillez consulter Désactivation d'un utilisateur IAM.

> **Warning**
>
> Une fois qu'un utilisateur IAM et ses clés d'accès sont supprimés, ils ne peuvent pas être restaurés ou récupérés.

Prérequis : affichage de l'accès de l'utilisateur IAM

Avant de supprimer un utilisateur, passez en revue ses activités récentes au niveau du service. Ceci permet d'éviter de supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour

de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Suppression d'un utilisateur IAM (console)

Lorsque vous utilisez le Console de gestion AWS pour supprimer un utilisateur IAM, IAM supprime automatiquement les informations associées suivantes :

- Identificateur de l'utilisateur IAM
- Toute appartenance à un groupe, ce qui signifie que l'utilisateur IAM est supprimé de tous les groupes dont il était membre
- Tous les mots de passe associés à utilisateur IAM
- Toutes les politiques en ligne intégrées à l'utilisateur IAM (politiques qui étaient appliquées à l'utilisateur IAM à l'aide des autorisations de groupe d'utilisateurs ne sont pas affectées)

> **Note**
>
> IAM supprime toutes les politiques gérées attachées à l'utilisateur IAM lorsque vous supprimez l'utilisateur, mais ne supprime pas les politiques gérées.

- Tous les dispositifs MFA associés

Pour supprimer un utilisateur IAM (console)

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d'accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation de gauche, sélectionnez Utilisateurs, puis cochez la case en regard du nom d'utilisateur IAM que vous souhaitez supprimer, pas le nom ou la ligne elle-même.
4. En haut de la page, sélectionnez Delete (Supprimer).

> **Note**
>
> Si certains utilisateurs disposent de clés d'accès actives, vous devez désactiver ces clés avant de supprimer les utilisateurs. Pour de plus amples informations, veuillez consulter Pour désactiver une clé d'accès pour un utilisateur IAM.

5. Dans la boîte de dialogue de confirmation, saisissez le nom d'utilisateur dans le champ de saisie de texte pour confirmer la suppression de l'utilisateur. Sélectionnez Delete (Supprimer).

La console affiche une notification d'état indiquant que l'utilisateur IAM a été supprimé.

Suppression d'un utilisateur IAM (AWS CLI)

Contrairement au Console de gestion AWS, lorsque vous supprimez un utilisateur IAM avec le AWS CLI, vous devez supprimer manuellement les éléments attachés à cet utilisateur IAM. La procédure suivante illustre ce processus.

Pour supprimer un utilisateur IAM de votre Compte AWS ()AWS CLI

1. Supprimez le mot de passe de l'utilisateur, s'il en a un.

   `aws iam delete-login-profile`
2. Supprimez les clés d'accès de l'utilisateur, si l'utilisateur en possède une.

   `aws iam list-access-keys` (pour répertorier les clés d'accès de l'utilisateur) et `aws iam delete-access-key`
3. Supprimez le certificat de signature de l'utilisateur. Notez que la suppression d'une information d'identification de sécurité est définitive et que vous ne pourrez plus récupérer celle-ci.

   `aws iam list-signing-certificates` (pour répertorier les certificats de signature de l'utilisateur) et `aws iam delete-signing-certificate`
4. Supprimez la clé publique SSH de l'utilisateur, s'il en a une.

   `aws iam list-ssh-public-keys` (pour répertorier les clés publiques SSH de l'utilisateur) et `aws iam delete-ssh-public-key`
5. Supprimez les informations d'identification Git.

`aws iam list-service-specific-credentials` (pour répertorier les informations d'identification git de l'utilisateur) et `aws iam delete-service-specific-credential`

6. Désactivez l'authentification multifacteur (MFA) de l'utilisateur, s'il en a une.

   `aws iam list-mfa-devices` (pour répertorier les dispositifs MFA de l'utilisateur), `aws iam deactivate-mfa-device` (pour désactiver le dispositif), et `aws iam delete-virtual-mfa-device` (pour supprimer définitivement un dispositif MFA virtuel)

7. Supprimez les politiques en ligne de l'utilisateur.

   `aws iam list-user-policies` (pour répertorier les politiques en ligne pour l'utilisateur) et `aws iam delete-user-policy` (pour supprimer la politique)

8. Détachez toutes les politiques gérées attachées à l'utilisateur.

   `aws iam list-attached-user-policies` (pour répertorier les politiques gérées attachées à l'utilisateur) et `aws iam detach-user-policy` (pour détacher la politique)

9. Supprimez l'utilisateur à partir de tous les groupes IAM.

   `aws iam list-groups-for-user` (pour répertorier les groupes IAM auxquels l'utilisateur appartient) et `aws iam remove-user-from-group`

10. Supprimez l'utilisateur.

    `aws iam delete-user`

Désactivation d'un utilisateur IAM

Vous pouvez désactiver un utilisateur IAM pendant qu'il est temporairement absent de votre entreprise. Vous pouvez laisser ses informations d'identification utilisateur IAM en place tout en bloquant son accès à AWS .

Pour désactiver un utilisateur, créez et attachez une politique pour lui refuser l'accès à AWS. Vous pourrez rétablir l'accès de l'utilisateur ultérieurement.

Voici deux exemples de politiques de refus que vous pouvez attacher à un utilisateur pour lui refuser l'accès.

La politique suivante n'inclut pas de limite de temps. Vous devez supprimer la politique pour rétablir l'accès de l'utilisateur.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*"
      }
   ]
}
```

La politique suivante inclut une condition qui commence la politique le 24 décembre 2024 à 23 h 59 (UTC) et la termine le 28 février 2025 à 23 h 59 (UTC).

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
          "DateGreaterThan": {"aws:CurrentTime": "2024-12-24T23:59:59Z"},
          "DateLessThan": {"aws:CurrentTime": "2025-02-28T23:59:59Z"}
          }
       }
   ]
}
```

Contrôlez l'accès des utilisateurs IAM au Console de gestion AWS

Les utilisateurs IAM autorisés qui se connectent à vous Compte AWS via le Console de gestion AWS peuvent accéder à vos AWS ressources. La liste suivante indique les manières dont vous pouvez

accorder aux utilisateurs IAM l'accès à vos Compte AWS ressources via le Console de gestion AWS. Il montre également comment les utilisateurs d'IAM peuvent accéder aux autres fonctionnalités du AWS compte via le AWS site Web.

> **Note**
>
> L'utilisation d'IAM n'induit aucuns frais.

Le Console de gestion AWS

Vous créez un mot de passe pour chaque utilisateur IAM devant accéder à l' Console de gestion AWS. Les utilisateurs accèdent à la console via votre page de Compte AWS connexion compatible IAM. Pour plus d'informations sur l'accès à la page de connexion, consultez Connexion à AWS dans le Guide de l'utilisateur Connexion à AWS . Pour plus d'informations sur la création de mots de passe, consultez Mots de passe utilisateur dans AWS.

Vous pouvez empêcher un utilisateur IAM d'accéder au Console de gestion AWS en supprimant son mot de passe. Cela les empêche de se connecter à l' Console de gestion AWS aide de leurs identifiants de connexion. Cela ne modifie pas ses autorisations ni ne l'empêche d'accéder à la console à l'aide d'un rôle endossé. Si l'utilisateur dispose de clés d'accès actives, elles continuent de fonctionner et autorisent l' AWS CLI accès via les Outils pour Windows PowerShell, l' AWS API ou l'Application AWS Console Mobile.

Vos AWS ressources, telles que les instances Amazon EC2, les compartiments Amazon S3, etc.

Même si vos utilisateurs IAM ont des mots de passe, ils doivent également disposer d'une autorisation leur permettant d'accéder à vos ressources AWS . Lorsque vous créez un utilisateur IAM, celui-ci ne dispose par défaut d'aucune autorisation. Pour accorder à vos utilisateurs IAM les autorisations dont ils ont besoin, vous leur attachez des politiques. Si plusieurs utilisateurs IAM effectuent les mêmes tâches avec les mêmes ressources, vous pouvez les affecter à un groupe. Affectez ensuite les autorisations à ce groupe. Pour plus d'informations sur la création d'utilisateurs et de groupes IAM, veuillez consulter la rubrique IAM identités. Pour plus d'informations sur l'utilisation de stratégies pour l'octroi d'autorisations, consultez Gestion des accès pour AWS resources.

AWS Forums de discussion

Tout le monde peut lire les messages publiés sur les Forums de discussion AWS. Les utilisateurs qui souhaitent publier des questions ou des commentaires AWS sur le forum de discussion

peuvent le faire en utilisant leur nom d'utilisateur. La première fois qu'un utilisateur publie AWS sur le forum de discussion, il est invité à saisir un surnom et une adresse e-mail. Seul cet utilisateur peut utiliser ce surnom dans les forums de AWS discussion.

Vos informations Compte AWS de facturation et d'utilisation

Vous pouvez autoriser les utilisateurs à accéder à vos informations Compte AWS de facturation et d'utilisation. Pour de plus amples informations, veuillez consulter Contrôle de l'accès à vos informations de facturation dans le Guide de l'utilisateur AWS Billing .

Informations Compte AWS de votre profil

Les utilisateurs ne peuvent pas accéder aux informations Compte AWS de votre profil.

Vos informations Compte AWS de sécurité

Les utilisateurs ne peuvent pas accéder à vos informations Compte AWS de sécurité.

> **Note**
>
> Les politiques IAM contrôlent l'accès, quelle que soit l'interface. Par exemple, vous pouvez fournir à un utilisateur un mot de passe pour accéder au Console de gestion AWS. Les politiques de cet utilisateur (ou de tout groupe auquel appartient l'utilisateur) contrôleraient ce que l'utilisateur peut faire dans le Console de gestion AWS. Vous pouvez également fournir à l'utilisateur des clés d' AWS accès pour effectuer des appels d'API à AWS. Les politiques contrôleraient les actions que l'utilisateur pourrait appeler via une bibliothèque ou un client qui utilise ces clés d'accès pour l'authentification.

Modification des autorisations pour un utilisateur IAM

Vous pouvez modifier les autorisations d'un utilisateur IAM de votre site en Compte AWS modifiant son appartenance à un groupe, en copiant les autorisations d'un utilisateur existant, en attachant des politiques directement à un utilisateur ou en définissant une limite d'autorisation. Une limite d'autorisations contrôle les autorisations maximum dont un utilisateur peut disposer. Les limites d'autorisations constituent une AWS fonctionnalité avancée.

Pour plus d'informations sur les autorisations requises pour modifier les autorisations d'un utilisateur, consultez Autorisations requises pour accéder aux autres ressources IAM.

Rubriques

- Afficher l'accès des utilisateurs
- Générer une politique basée sur l'activité d'accès d'un utilisateur
- Ajout d'autorisations à un utilisateur (console)
- Modification des autorisations pour un utilisateur (console)
- Pour supprimer une politique d'autorisations d'un utilisateur (console)
- Pour supprimer la limite des autorisations d'un utilisateur (console)
- Ajouter et supprimer les autorisations (AWS CLI ou AWS API) d'un utilisateur

Afficher l'accès des utilisateurs

Avant de modifier les autorisations d'un utilisateur, vous devez passer en revue ses activités récentes au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Générer une politique basée sur l'activité d'accès d'un utilisateur

Vous pouvez parfois octroyer plus d'autorisations à une entité IAM (utilisateur ou rôle) qu'elle n'en a besoin. Pour affiner les autorisations que vous octroyez, vous pouvez générer une politique IAM basée sur l'activité d'accès d'une entité. IAM Access Analyzer examine vos AWS CloudTrail journaux et génère un modèle de politique contenant les autorisations qui ont été utilisées par l'entité dans la plage de dates que vous avez spécifiée. Vous pouvez utiliser le modèle pour créer une politique gérée avec des autorisations affinées, puis l'attacher à l'entité IAM. Ainsi, vous accordez uniquement les autorisations dont l'utilisateur ou le rôle a besoin pour interagir avec les AWS ressources correspondant à votre cas d'utilisation spécifique. Pour en savoir plus, consultez Génération d'une politique de l'analyseur d'accès IAM.

Ajout d'autorisations à un utilisateur (console)

IAM propose trois méthodes pour ajouter des politiques d'autorisations à un utilisateur :

- Ajouter l'utilisateur IAM à un groupe IAM : faites de l'utilisateur un membre d'un groupe. Les politiques du groupe sont attachées à l'utilisateur.
- Copier les autorisations d'un utilisateur IAM existant : copiez toutes les appartenances à un groupe, toutes les politiques gérées attachées, les politiques en ligne et toutes les limites des autorisations existantes d'un utilisateur source.

- Attacher directement des politiques à l'utilisateur IAM : attachez directement une politique gérée à l'utilisateur. Pour faciliter la gestion des autorisations, affectez vos politiques à un groupe, puis faites de ces utilisateurs IAM des membres des groupes appropriés.

> **Important**
>
> Si l'utilisateur dispose d'une limite des autorisations, alors vous ne pouvez pas ajouter à l'utilisateur plus d'autorisations que le permet la limite.

Pour ajouter des autorisations en ajoutant l'utilisateur IAM à un groupe

L'ajout d'un utilisateur IAM à un groupe IAM met immédiatement à jour les autorisations de l'utilisateur avec les autorisations définies pour le groupe.

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d'accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez utilisateurs.
4. Dans la liste des Users (Utilisateurs), choisissez le nom d'utilisateur IAM.
5. Sélectionnez l'onglet Groupes pour afficher la liste des groupes qui incluent l'utilisateur actuel.
6. Choisissez Ajouter l'utilisateur aux groupes.
7. Activez la case à cocher pour chaque groupe que l'utilisateur doit rejoindre. La liste affiche le nom de chaque groupe et les politiques que l'utilisateur reçoit s'il devient membre de ce groupe.
8. (Facultatif) Vous pouvez choisir Créer un groupe pour définir un nouveau groupe. Cela est utile si vous souhaitez ajouter l'utilisateur à un groupe auquel sont associées des politiques différentes de celles des groupes existants :

    a. Dans le nouvel onglet, pour Nom du groupe d'utilisateurs, saisissez le nom de votre nouveau groupe.

> **Note**
>
> Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas. Les noms de groupe peuvent combiner jusqu'à 128 lettres, chiffres et caractères suivants : plus (+), égal (=), virgule (,), point (.), arobase (@) et tiret (-). Les noms doivent être uniques au sein d'un compte. Ils ne sont pas distingués au cas par cas. Par exemple, vous ne pouvez pas créer deux groupes nommés TESTGROUP et testgroup.

b. Cochez une ou plusieurs cases pour les politiques gérées que vous souhaitez attacher au groupe. Vous pouvez également créer une politique gérée en choisissant Créer une politique. Le cas échéant, revenez dans la fenêtre ou l'onglet du navigateur une fois la nouvelle politique créée. Choisissez Refresh (Actualiser), puis choisissez la nouvelle politique à attacher à votre groupe. Pour de plus amples informations, veuillez consulter Définition d'autorisations IAM personnalisées avec des politiques gérées par le client.

c. Choisissez Créer un groupe d'utilisateurs.

d. Revenez à l'onglet initial, actualisez votre liste de groupes. Puis cochez la case correspondant à votre nouveau groupe.

9. Choisissez Ajouter l'utilisateur au(x) groupe(s).

La console affiche un message d'état vous informant que l'utilisateur a été ajouté aux groupes que vous avez spécifié.

Pour ajouter des autorisations en copiant celles d'un autre utilisateur IAM

Si vous choisissez d'ajouter des autorisations à un utilisateur IAM en copiant des autorisations, IAM copie toutes les appartenances à des groupes, les politiques gérées attachées, les politiques en ligne et toutes les limites des autorisations existantes de l'utilisateur spécifié et les applique immédiatement à l'utilisateur actuellement sélectionné.

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .

2. Sur la page d’accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez utilisateurs.
4. Dans la liste des Users (Utilisateurs), choisissez le nom d'utilisateur IAM.
5. Sous l’onglet Autorisations, choisissez Ajouter des autorisations.
6. Sur la page Ajouter des autorisations, choisissez Copier des autorisations. La liste affiche les utilisateurs IAM disponibles ainsi que leur appartenance à un groupe et les politiques attachées.
7. Sélectionnez le bouton radio en regard de l'utilisateur dont vous voulez copier les autorisations.
8. Choisissez Suivant pour afficher la liste des modifications apportées à l'utilisateur. Choisissez ensuite Add permissions (Ajouter des autorisations).

La console affiche un message d’état vous informant que les autorisations ont été copiées depuis l’utilisateur IAM que vous avez spécifié.

Pour ajouter des autorisations en attachant les politiques directement à l’utilisateur IAM

Vous pouvez attacher une politique gérée directement à un utilisateur IAM. Les autorisations mises à jour sont appliquées immédiatement.

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d’accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez utilisateurs.
4. Dans la liste des Users (Utilisateurs), choisissez le nom d'utilisateur IAM.
5. Sous l’onglet Autorisations, choisissez Ajouter des autorisations.
6. Sur la page Ajouter des autorisations, choisissez Attacher directement les politiques. La liste des Politiques d’autorisations affiche les politiques disponibles ainsi que leurs types de politiques et les entités associées.
7. Sélectionnez le bouton en regard du Nom de la politique que vous souhaitez attacher.

8. Choisissez Suivant pour afficher la liste des modifications apportées à l'utilisateur. Choisissez ensuite Add permissions (Ajouter des autorisations).

   La console affiche un message d’état vous informant que la politique a été ajoutée à l’utilisateur IAM que vous avez spécifié.

Pour définir la limite des autorisations pour un utilisateur IAM

Une limite d'autorisations est une fonctionnalité avancée de gestion des AWS autorisations utilisée pour définir le maximum d'autorisations qu'un utilisateur IAM peut avoir. La définition d’une limite des autorisations restreint immédiatement les autorisations de l’utilisateur IAM à cette limite, quelles que soient les autres autorisations accordées.

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d’accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez utilisateurs.
4. Dans la liste Utilisteurs, choisissez le nom de l’utilisateur IAM dont vous souhaitez modifier la limite des autorisations.
5. Sélectionnez l’onglet Autorisations. Si nécessaire, ouvrez la section Limite d'autorisations, puis choisissez Définir une limite d'autorisations.
6. Sur la page Définir la limite des autorisations, sous Politiques d’autorisations, sélectionnez la politique que vous souhaitez utiliser pour la limite des autorisations.
7. Choisissez Set boundary (Définir une limite).

   La console affiche un message d’état vous informant que la limite des autorisations a été ajoutée.

Modification des autorisations pour un utilisateur (console)

IAM vous permet de modifier les autorisations associées à un utilisateur de la manière suivante :

- Edit a permissions policy (Modifier une politique d'autorisations) : modifiez la politique en ligne d'un utilisateur, la politique en ligne du groupe de l'utilisateur ou une politique gérée attachée

directement à l'utilisateur ou à partir d'un groupe. Si l'utilisateur dispose d'une limite d'autorisations, alors vous ne pouvez pas fournir plus d'autorisations que le permet la politique utilisée comme limite d'autorisations de l'utilisateur.

- Changing the permissions boundary (Modification de la limite d'autorisations) : modifiez la politique utilisée comme limite d'autorisations pour l'utilisateur. Cette action peut développer ou limiter les autorisations maximum dont dispose un utilisateur.

Modification d'une politique d'autorisations attachée à un utilisateur

La modification d'autorisations met immédiatement à jour l'accès de l'utilisateur.

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d'accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez utilisateurs.
4. Dans la liste Utilisteurs, choisissez le nom de l'utilisateur IAM dont vous souhaitez modifier la limite des autorisations.
5. Sélectionnez l'onglet Autorisations. Si nécessaire, ouvrez la section Limite des autorisations.
6. Choisissez le nom de la politique que vous souhaitez modifier pour en afficher les détails. Choisissez l'onglet Entités attachées pour afficher d'autres entités (utilisateurs, groupes et rôles IAM) qui pourraient être affectées par la modification de la politique.
7. Choisissez l'onglet Autorisations et examinez les autorisations accordées par la politique. Pour modifier les autorisations, choisissez Modifier.
8. Modifiez la politique et résolvez les recommandations sur la validation des politiques. Pour de plus amples informations, veuillez consulter Modification de politiques IAM.
9. Choisissez Suivant, examinez le récapitulatif de la politique, puis choisissez Enregistrer les modifications.

La console affiche un message d'état vous informant que la politique a été mise à jour.

Pour changer la limite des autorisations pour un utilisateur

La modification d’une limite des autorisations met immédiatement à jour l’accès de l’utilisateur.

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d’accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez utilisateurs.
4. Dans la liste Utilisteurs, choisissez le nom de l’utilisateur IAM dont vous souhaitez modifier la limite des autorisations.
5. Sélectionnez l’onglet Autorisations. Si nécessaire, ouvrez la section Permissions boundary (Limite d'autorisations), puis choisissez Change boundary (Modifier une limite).
6. Sélectionnez la politique que vous souhaitez utiliser pour la limite d'autorisations.
7. Choisissez Set boundary (Définir une limite).

   La console affiche un message d’état vous informant que la limite des autorisations a été modifiée.

Pour supprimer une politique d’autorisations d’un utilisateur (console)

La suppression d’autorisations met immédiatement à jour l’accès de l’utilisateur.

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d’accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez utilisateurs.
4. Choisissez le nom de l’utilisateur dont vous souhaitez supprimer les politiques d’autorisations.
5. Sélectionnez l’onglet Autorisations.

6. Si vous souhaitez supprimer les autorisations en supprimant une politique existante, affichez la colonne Attachée via pour comprendre comment l’utilisateur obtient cette politique avant de choisir Supprimer pour supprimer celle-ci :

   - Si la politique s'applique en raison de l'appartenance à un groupe, choisissez Supprimer pour supprimer l'utilisateur du groupe. N'oubliez pas que vous pouvez avoir plusieurs politiques attachées à un seul groupe. Si vous supprimez un utilisateur d'un groupe, il perd l'accès à toutes les politiques qu'il reçoit par le biais de cette appartenance au groupe.
   - Si la politique est une politique gérée attachée directement à l'utilisateur, choisissez Supprimer pour détacher la politique de l'utilisateur. Cela n'affecte pas la politique elle-même ni aucune autre entité à laquelle la politique pourrait être attachée.
   - S’il s’agit d’une politique en ligne intégrée, sélectionnez Supprimer pour supprimer la politique d’IAM. Les politiques en ligne attachées directement à un utilisateur existent uniquement sur cet utilisateur.

   Si la politique a été accordée à l’utilisateur par le biais d’une adhésion à un groupe, la console affiche un message d’état vous informant que l’utilisateur IAM a été supprimé du groupe IAM. Si la politique est directement attachée ou intégrée, le message d’état vous informe que la politique a été supprimée.

Pour supprimer la limite des autorisations d’un utilisateur (console)

La suppression de la limite des autorisations met immédiatement à jour l’accès de l’utilisateur.

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d’accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez utilisateurs.
4. Dans la liste Utilisteurs, choisissez le nom de l’utilisateur IAM dont vous souhaitez supprimer la limite des autorisations.
5. Sélectionnez l’onglet Autorisations. Si nécessaire, ouvrez la section Limite des autorisations.

6. Choisissez Change boundary (Modifier une limite). Pour confirmer que vous souhaitez supprimer la limite des autorisations, sélectionnez Supprimer la limite dans la boîte de dialogue de confirmation.

La console affiche un message d'état vous informant que la limite des autorisations a été supprimée.

Ajouter et supprimer les autorisations (AWS CLI ou AWS API) d'un utilisateur

Pour ajouter ou supprimer des autorisations par programme, vous devez ajouter ou supprimer l'appartenance au groupe, attacher ou détacher les politiques gérées, ou ajouter ou supprimer les politiques en ligne. Pour plus d'informations, consultez les rubriques suivantes :

- Modification des utilisateurs dans les groupes IAM
- Ajout et suppression d'autorisations basées sur l'identité IAM

Mots de passe utilisateur dans AWS

Vous pouvez gérer les mots de passe des utilisateurs IAM de votre compte. Les utilisateurs IAM ont besoin de mots de passe pour accéder au Console de gestion AWS. Les utilisateurs n'ont pas besoin de mots de passe pour accéder aux AWS ressources par programmation à l' AWS CLI aide des outils pour Windows PowerShell ou. AWS SDKs APIs Pour ces environnements, vous avez la possibilité d'attribuer des clés d'accès aux utilisateurs IAM. Cependant, il existe d'autres alternatives plus sécurisées aux clés d'accès que nous vous recommandons d'envisager d'abord. Pour de plus amples informations, veuillez consulter AWS informations d'identification de sécurité.

> **Note**
>
> Si l'un de vos utilisateurs IAM perd ou oublie son mot de passe, vous ne pouvez pas le récupérer auprès d'IAM. En fonction de vos paramètres, l'utilisateur ou l'administrateur doit générer un nouveau mot de passe.

Table des matières

- Définition d'une politique de mot de passe du compte pour les utilisateurs IAM
- Gérer les mots de passe des utilisateurs IAM

- Octroi aux utilisateurs IAM de l’autorisation de modification de leurs propres mots de passe
- Modification par l'utilisateur IAM de son propre mot de passe

Définition d’une politique de mot de passe du compte pour les utilisateurs IAM

Vous pouvez définir une politique de mot de passe personnalisée Compte AWS pour définir les exigences de complexité et les périodes de rotation obligatoires pour les mots de passe de vos utilisateurs IAM. Si vous ne définissez pas de politique de mot de passe personnalisée, les mots de passe des utilisateurs IAM doivent respecter la politique de AWS mot de passe par défaut. Pour de plus amples informations, veuillez consulter Options de politique de mot de passe personnalisé.

Rubriques

- Règles relatives à la définition d'une politique de mot de passe
- Autorisations requises pour définir une politique de mot de passe
- Politique de mot de passe par défaut
- Options de politique de mot de passe personnalisé
- Pour définir une politique de mot de passe (console)
- Pour modifier une politique de mot de passe (console)
- Pour supprimer une politique de mot de passe personnalisée (console)
- Définition d'une politique de mot de passe (AWS CLI)
- Définition d'une politique de mot de passe (AWS API)

Règles relatives à la définition d'une politique de mot de passe

La politique de mot de passe IAM ne s'applique pas au Utilisateur racine d'un compte AWS mot de passe ou aux clés d'accès utilisateur IAM. Si un mot de passe expire, l'utilisateur IAM ne peut pas se connecter Console de gestion AWS mais peut continuer à utiliser ses clés d'accès.

Lorsque vous créez ou modifiez une politique de mot de passe, la plupart de ses paramètres sont appliqués la fois suivante où vos utilisateurs modifient leurs mots de passe. Toutefois, certains des paramètres sont appliqués immédiatement. Par exemple :

- Lorsque les exigences de longueur minimale et de type de caractères sont modifiés, les nouveaux paramètres sont appliqués à la prochaine modification des mots de passe. Les utilisateurs ne sont pas contraints à modifier leurs mots de passe, même si ceux-ci ne respectent pas la politique de mot de passe modifiée.

- Lorsque vous définissez une période d'expiration de mot de passe, celle-ci est appliquée immédiatement. Par exemple, supposons que vous définissez une période d'expiration de mot de passe de 90 jours. Dans ce cas, le mot de passe expire pour tous les utilisateurs IAM dont le mot de passe existant date de plus de 90 jours. Ces utilisateurs sont tenus de modifier leur mot de passe la première fois qu'ils se connectent.

Vous ne pouvez pas créer de « politique de verrouillage » pour empêcher l'accès d'un utilisateur à un compte après un nombre défini de tentatives de connexion ayant échoué. Pour renforcer votre sécurité, nous vous recommandons de combiner des politiques de mot de passe d'un niveau de sécurité élevé à la Multi-Factor Authentication (MFA). Pour plus d'informations sur l'authentification MFA, consultez AWS Multi-factor authentification dans IAM.

Autorisations requises pour définir une politique de mot de passe

Vous devez configurer les autorisations pour permettre à une entité IAM (utilisateur ou rôle) d'afficher ou de modifier sa politique de mot de passe de compte. Vous pouvez inclure les actions de politique de mot de passe suivantes dans une politique IAM :

- `iam:GetAccountPasswordPolicy` : permet à l'entité d'afficher la politique de mot de passe pour son compte
- `iam:DeleteAccountPasswordPolicy` : permet à l'entité de supprimer la politique de mot de passe personnalisée pour son compte et de revenir à la politique de mot de passe par défaut
- `iam:UpdateAccountPasswordPolicy` : permet à l'entité de créer ou de modifier la politique de mot de passe personnalisée pour son compte

La politique suivante permet un accès complet pour afficher et modifier la politique de mot de passe du compte. Pour apprendre à créer une politique IAM à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessPasswordPolicy",
            "Effect": "Allow",
            "Action": [
```

```
                "iam:GetAccountPasswordPolicy",
                "iam:DeleteAccountPasswordPolicy",
                "iam:UpdateAccountPasswordPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

Pour plus d'informations sur les autorisations requises par les utilisateurs IAM pour modifier leur propre mot de passe, veuillez consulter Octroi aux utilisateurs IAM de l'autorisation de modification de leurs propres mots de passe.

Politique de mot de passe par défaut

Si un administrateur ne définit pas de politique de mot de passe personnalisée, les mots de passe des utilisateurs IAM doivent respecter la politique de AWS mot de passe par défaut.

La politique de mot de passe par défaut exige les conditions suivantes :

- Longueur minimale du mot de passe de 8 caractères et longueur maximale de 128 caractères
- Au minimum trois des types de caractères suivants : majuscules, minuscules, chiffres et les caractères non alphanumériques (`! @ # $ % ^ & * ( ) _ + - = [ ] { } | '`)
- Ne pas être identique à votre Compte AWS nom ou à votre adresse e-mail
- Mot de passe sans expiration

Options de politique de mot de passe personnalisé

Lorsque vous configurez une politique de mot de passe personnalisée pour votre compte, vous pouvez spécifier les conditions suivantes :

- Password minimum length (Longueur minimale du mot de passe) : vous pouvez spécifier une longueur minimale de 6 caractères et une longueur maximale de 128 caractères.
- Niveau de sécurité du mot de passe : vous pouvez cocher l'une des cases suivantes pour définir le niveau de sécurité de vos mots de passe utilisateur IAM :
  - Exiger au moins une lettre majuscule de l'alphabet latin (A–Z)
  - Exiger au moins une lettre minuscule de l'alphabet latin (a–z)
  - Nécessite au moins un chiffre

  - Exiger au moins un caractère non alphanumérique `! @ # $ % ^ & * ( ) _ + - = [ ] { } | '`
- Turn on password expiration (Activer l'expiration du mot de passe) : vous pouvez sélectionner et spécifier une durée minimale de 1 jour et une durée maximale de 1 095 jours pendant laquelle les mots de passe utilisateur IAM sont valides après leur définition. Par exemple, si vous spécifiez un délai d'expiration de 90 jours, cela a un impact immédiat sur tous vos utilisateurs. Pour les utilisateurs dont le mot de passe date de plus de 90 jours, ils doivent définir un nouveau mot de passe lorsqu'ils se connectent à la console après la modification. Les utilisateurs dont le mot de passe est vieux de 75 à 89 jours reçoivent un Console de gestion AWS avertissement concernant l'expiration de leur mot de passe. Les utilisateurs IAM peuvent modifier leur mot de passe à tout moment s'ils en ont l'autorisation. Lorsqu'ils définissent un nouveau mot de passe, la date d'expiration est réinitialisée. Un utilisateur IAM ne peut avoir qu'un mot de passe valide à la fois.
- L'expiration du mot de passe nécessite une réinitialisation par l'administrateur : sélectionnez cette option pour empêcher les utilisateurs IAM de l'utiliser Console de gestion AWS pour mettre à jour leur propre mot de passe après son expiration. Avant de sélectionner cette option, vérifiez que votre Compte AWS a plusieurs utilisateurs disposant des autorisations d'administrateur nécessaires pour réinitialiser les mots de passe utilisateur IAM. Les administrateurs disposant de l'autorisation `iam:UpdateLoginProfile` peuvent réinitialiser les mots de passe des utilisateurs IAM. Les utilisateurs IAM disposant de l'autorisation `iam:ChangePassword` et de clés d'accès actives peuvent réinitialiser leur propre mot de passe de console utilisateur IAM de manière programmatique. Si vous décochez cette case, les utilisateurs IAM dont les mots de passe ont expiré doivent tout de même définir un nouveau mot de passe avant de pouvoir accéder à l'Console de gestion AWS.
- Allow users to change their own password (Autoriser les utilisateurs à modifier leur propre mot de passe) : vous pouvez permettre à tous les utilisateurs IAM de votre compte d'utiliser la console IAM pour modifier leur mot de passe. Cela permet aux utilisateurs d'accéder à l'action `iam:ChangePassword` uniquement pour leur propre utilisateur et à l'action `iam:GetAccountPasswordPolicy`. Cette option n'associe aucune politique d'autorisations à chaque utilisateur. IAM applique plutôt les autorisations au niveau du compte pour tous les utilisateurs. Vous pouvez également n'autoriser que certains utilisateurs à gérer leurs propres mots de passe. Pour ce faire, décochez cette case. Pour plus d'informations sur l'utilisation de politiques visant à limiter les utilisateurs pouvant gérer les mots de passe, consultez la section Octroi aux utilisateurs IAM de l'autorisation de modification de leurs propres mots de passe.
- Prevent password reuse (Empêcher la réutilisation d'un mot de passe) : vous pouvez empêcher les utilisateurs IAM de réutiliser un certain nombre de mots de passe précédents. Vous pouvez

spécifier le nombre de mots de passe précédents qui ne peuvent pas être réutilisés, entre 1 et 24 mots de passe.

Pour définir une politique de mot de passe (console)

Vous pouvez utiliser le Console de gestion AWS pour créer, modifier ou supprimer une politique de mot de passe personnalisée. Les modifications apportées à la politique de mot de passe s’appliquent aux nouveaux utilisateurs IAM créés après cette modification de politique et aux utilisateurs IAM existants lorsqu’ils modifient leur mot de passe.

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d’accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez Paramètres du compte.
4. Dans la section Password policy (Politique de mot de passe), sélectionnez Edit (Modifier).
5. Choisissez Custom (Personnalisé) pour utiliser une politique de mot de passe personnalisée.
6. Sélectionnez les options que vous souhaitez appliquer à votre politique de mot de passe, puis sélectionnez Enregistrer les modifications.
7. Confirmez que vous souhaitez définir la politique de mot de passe personnalisée en sélectionnant Définir personnalisé.

La console affiche un message d’état vous informant que les exigences en matière de mot de passe pour les utilisateurs IAM ont été mises à jour.

Pour modifier une politique de mot de passe (console)

Vous pouvez utiliser le Console de gestion AWS pour créer, modifier ou supprimer une politique de mot de passe personnalisée. Les modifications apportées à la politique de mot de passe s’appliquent aux nouveaux utilisateurs IAM créés après cette modification de politique et aux utilisateurs IAM existants lorsqu’ils modifient leur mot de passe.

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d’accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez Paramètres du compte.
4. Dans la section Password policy (Politique de mot de passe), sélectionnez Edit (Modifier).
5. Sélectionnez les options que vous souhaitez appliquer à votre politique de mot de passe, puis sélectionnez Enregistrer les modifications.
6. Confirmez que vous souhaitez définir la politique de mot de passe personnalisée en sélectionnant Définir personnalisé.

   La console affiche un message d’état vous informant que les exigences en matière de mot de passe pour les utilisateurs IAM ont été mises à jour.

Pour supprimer une politique de mot de passe personnalisée (console)

Vous pouvez utiliser le Console de gestion AWS pour créer, modifier ou supprimer une politique de mot de passe personnalisée. Les modifications apportées à la politique de mot de passe s’appliquent aux nouveaux utilisateurs IAM créés après cette modification de politique et aux utilisateurs IAM existants lorsqu’ils modifient leur mot de passe.

Console

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d’accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez Paramètres du compte.
4. Dans la section Password policy (Politique de mot de passe), sélectionnez Edit (Modifier).
5. Choisissez IAM default (IAM par défaut) pour supprimer la politique de mot de passe personnalisée, puis Save changes (Enregistrer les modifications).
6. Confirmez que vous souhaitez définir la politique de mot de passe IAM par défaut en sélectionnant Définir personnalisé.

La console affiche un message d'état vous informant que la politique de mot de passe est définie sur le réglage par défaut d'IAM.

Définition d'une politique de mot de passe (AWS CLI)

Vous pouvez utiliser le AWS Command Line Interface pour définir une politique de mot de passe.

Pour gérer la politique de mot de passe de compte personnalisée depuis le AWS CLI

Exécutez les commandes suivantes :

- Pour créer ou modifier la politique de mot de passe personnalisée : `aws iam update-account-password-policy`
- Pour afficher la politique de mot de passe : `aws iam get-account-password-policy`
- Pour supprimer la politique de mot de passe personnalisée : `aws iam delete-account-password-policy`

Définition d'une politique de mot de passe (AWS API)

Vous pouvez utiliser les opérations de AWS l'API pour définir une politique de mot de passe.

Pour gérer la politique de mot de passe de compte personnalisée à partir de l' AWS API

Appelez les opérations suivantes :

- Pour créer ou modifier la politique de mot de passe personnalisée : `UpdateAccountPasswordPolicy`
- Pour afficher la politique de mot de passe : `GetAccountPasswordPolicy`
- Pour supprimer la politique de mot de passe personnalisée : `DeleteAccountPasswordPolicy`

Gérer les mots de passe des utilisateurs IAM

Les utilisateurs IAM qui utilisent le Console de gestion AWS pour utiliser des AWS ressources doivent disposer d'un mot de passe pour se connecter. Vous pouvez créer, modifier ou supprimer un mot de passe pour un utilisateur IAM dans votre compte AWS .

Après avoir attribué un mot de passe à un utilisateur, celui-ci peut se connecter à l' Console de gestion AWS aide de l'URL de connexion de votre compte, qui ressemble à ceci :

```
https://12-digit-AWS-account-ID or alias.signin.aws.amazon.com/console
```

Pour plus d'informations sur la manière dont les utilisateurs IAM se connectent au Console de gestion AWS, consultez Comment se connecter AWS dans le Guide de l'Connexion à AWS utilisateur.

Même si vos utilisateurs ont leurs propres mots de passe, ils toujours besoin de votre autorisation pour accéder à vos ressources AWS . Par défaut, un utilisateur ne dispose d'aucune autorisation. Pour accorder à vos utilisateurs les autorisations dont ils ont besoin, vous leur attribuez des politiques ou aux groupes dont ils font partie. Pour plus d'informations sur la création d'utilisateurs et de groupes, consultez IAM identités. Pour plus d'informations sur l'utilisation de stratégies pour l'octroi d'autorisations, consultez Modification des autorisations pour un utilisateur IAM.

Vous pouvez accorder aux utilisateurs l'autorisation de modifier leurs propres mots de passe. Pour de plus amples informations, veuillez consulter Octroi aux utilisateurs IAM de l'autorisation de modification de leurs propres mots de passe. Pour plus d'informations sur la façon dont les utilisateurs accèdent à la page de connexion à votre compte, consultez Connexion à AWS dans le Guide de l'utilisateur Connexion à AWS .

Rubriques

- Création, modification ou suppression d'un mot de passe utilisateur IAM (console)

Création, modification ou suppression d'un mot de passe utilisateur IAM (console)

Vous pouvez utiliser le Console de gestion AWS pour gérer les mots de passe de vos utilisateurs IAM.

Les besoins de vos utilisateurs en matière d'accès peuvent changer au fil du temps. Vous devrez peut-être autoriser un utilisateur destiné à accéder à la CLI à accéder à la console, modifier le mot de passe d'un utilisateur car il reçoit un e-mail contenant ses informations d'identification, ou supprimer un utilisateur lorsqu'il quitte votre organisation ou n'a plus besoin d' AWS accès.

Pour créer le mot de passe d'un utilisateur IAM (console)

Utilisez cette procédure pour donner accès à une console utilisateur en générant un mot de passe associé au nom d'utilisateur.

Console

1. Suivez la procédure de connexion adaptée à votre type d'utilisateur, comme décrit dans la rubrique Comment se connecter AWS dans le Guide de l'AWS Sign-In utilisateur.
2. Sur la page d'accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez utilisateurs.
4. Choisissez le nom de l'utilisateur dont vous souhaitez créer le mot de passe.
5. Cliquez sur l'onglet Informations d'identification de sécurité, puis sous Connexion à la console, sélectionnez Activer l'accès à la console.
6. Dans la boîte de dialogue Activer l'accès à la console, sélectionnez Réinitialiser le mot de passe, puis choisissez si vous préférez qu'IAM génère un mot de passe ou si vous voulez en générer un personnalisé :

   - Pour qu'IAM génère un mot de passe, sélectionnez Autogenerated password (Mot de passe généré automatiquement).
   - Pour créer un mot de passe personnalisé, choisissez Custom password (Mot de passe personnalisé), puis tapez le mot de passe.

   > **Note**
   >
   > Le mot de passe que vous créez doit respecter la politique de mot de passe du compte.

7. Pour obliger l'utilisateur à générer un nouveau mot de passe au moment de la connexion, sélectionnez L'utilisateur doit générer un nouveau mot de passe à la prochaine connexion.
8. Pour demander à l'utilisateur d'utiliser le nouveau mot de passe immédiatement, sélectionnez Révoquer les sessions de console actives. Cela associe une politique en ligne à l'utilisateur IAM qui lui refuse l'accès aux ressources si ses informations d'identification sont antérieures à la date spécifiée par la politique.
9. Choisissez Réinitialiser le mot de passe.
10. La boîte de dialogue Mot de passe de la console vous informe que vous avez activé le nouveau mot de passe de l'utilisateur. Pour afficher le mot de passe afin de le partager avec l'utilisateur, choisissez Afficher dans la boîte de dialogue Mot de passe de la console. Sélectionnez Télécharger le fichier .csv pour télécharger un fichier contenant les informations d'identification de l'utilisateur.

> **Important**
>
> Pour des raisons de sécurité, vous ne pouvez pas accéder au mot de passe après avoir effectué cette étape, mais vous pouvez créer un nouveau mot de passe à tout moment.

La console affiche un message d'état vous informant que l'accès à la console a été activé.

Pour changer le mot de passe d'un utilisateur IAM (console)

Utilisez cette procédure pour mettre à jour un mot de passe associé au nom d'utilisateur.

Console

1. Suivez la procédure de connexion adaptée à votre type d'utilisateur, comme décrit dans la rubrique Comment se connecter AWS dans le Guide de l'AWS Sign-In utilisateur.
2. Sur la page d'accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez utilisateurs.
4. Choisissez le nom de l'utilisateur dont vous souhaitez changer le mot de passe.
5. Cliquez sur l'onglet Informations d'identification de sécurité, puis sous Connexion à la console, sélectionnez Gérer l'accès à la console.
6. Dans la boîte de dialogue Gérer l'accès à la console, sélectionnez Réinitialiser le mot de passe, puis choisissez si vous préférez qu'IAM génère un mot de passe ou si vous voulez générer un personnalisé :
   - Pour qu'IAM génère un mot de passe, sélectionnez Autogenerated password (Mot de passe généré automatiquement).
   - Pour créer un mot de passe personnalisé, choisissez Custom password (Mot de passe personnalisé), puis tapez le mot de passe.

> **Note**
>
> Le mot de passe que vous créez doit respecter la politique de mot de passe du compte.

7. Pour obliger l'utilisateur à générer un nouveau mot de passe au moment de la connexion, sélectionnez L'utilisateur doit générer un nouveau mot de passe à la prochaine connexion.
8. Pour demander à l'utilisateur d'utiliser le nouveau mot de passe immédiatement, sélectionnez Révoquer les sessions de console actives. Cela associe une politique en ligne à l'utilisateur IAM qui lui refuse l'accès aux ressources si ses informations d'identification sont antérieures à la date spécifiée par la politique.
9. Choisissez Réinitialiser le mot de passe.
10. La boîte de dialogue Mot de passe de la console vous informe que vous avez activé le nouveau mot de passe de l'utilisateur. Pour afficher le mot de passe afin de le partager avec l'utilisateur, choisissez Afficher dans la boîte de dialogue Mot de passe de la console. Sélectionnez Télécharger le fichier .csv pour télécharger un fichier contenant les informations d'identification de l'utilisateur.

> **Important**
>
> Pour des raisons de sécurité, vous ne pouvez pas accéder au mot de passe après avoir effectué cette étape, mais vous pouvez créer un nouveau mot de passe à tout moment.

La console affiche un message d'état vous informant que l'accès à la console a été mis à jour.

Supprimer (désactiver) le mot de passe d'un utilisateur IAM (console)

Utilisez cette procédure pour supprimer un mot de passe associé au nom d'utilisateur, privant ainsi l'utilisateur de l'accès à la console.

> **Important**
>
> Vous pouvez empêcher un utilisateur IAM d'accéder au Console de gestion AWS en supprimant son mot de passe. Cela les empêche de se connecter à l' Console de gestion

AWS aide de leurs informations de connexion. Cela ne modifie pas ses autorisations ni ne l'empêche d'accéder à la console à l'aide d'un rôle endossé. Si l'utilisateur dispose de clés d'accès actives, celles-ci continuent de fonctionner et autorisent l'accès via les AWS CLI outils pour Windows PowerShell, AWS l'API ou l'application mobile de la AWS console.

Console

1. Suivez la procédure de connexion adaptée à votre type d'utilisateur, comme décrit dans la rubrique Comment se connecter AWS dans le Guide de l'AWS Sign-In utilisateur.
2. Sur la page d'accueil de la console IAM, dans le volet de navigation de gauche, saisissez votre requête dans la zone de texte Rechercher sur IAM.
3. Dans le panneau de navigation, choisissez utilisateurs.
4. Choisissez le nom de l'utilisateur dont vous souhaitez supprimer le mot de passe.
5. Cliquez sur l'onglet Informations d'identification de sécurité, puis sous Connexion à la console, sélectionnez Gérer l'accès à la console.
6. Pour demander à l'utilisateur de cesser immédiatement d'utiliser la console, sélectionnez Révoquer les sessions de console actives. Cela associe une politique en ligne à l'utilisateur IAM qui lui refuse l'accès aux ressources si ses informations d'identification sont antérieures à la date spécifiée par la politique.
7. Choisissez Désactiver l'accès

La console affiche un message d'état vous informant que l'accès à la console a été désactivé.

Création, modification ou suppression d'un mot de passe utilisateur IAM (AWS CLI)

Vous pouvez utiliser le AWS CLI pour gérer les mots de passe de vos utilisateurs IAM.

Pour créer un mot de passe (AWS CLI)

1. (Facultatif) Pour déterminer si un utilisateur possède un mot de passe, exécutez cette commande : aws iam get-login-profile
2. Pour créer un mot de passe, exécutez cette commande : aws iam create-login-profile

Pour modifier le mot de passe d'un utilisateur (AWS CLI)

1. (Facultatif) Pour déterminer si un utilisateur possède un mot de passe, exécutez cette commande : aws iam get-login-profile
2. Pour modifier un mot de passe, exécutez cette commande : aws iam update-login-profile

Pour supprimer (désactiver) le mot de passe d'un utilisateur (AWS CLI)

1. (Facultatif) Pour déterminer si un utilisateur possède un mot de passe, exécutez cette commande : aws iam get-login-profile
2. (Facultatif) Pour déterminer quand un mot de passe a été utilisé pour la dernière fois, exécutez cette commande : aws iam get-user
3. Pour supprimer un mot de passe, exécutez cette commande : aws iam delete-login-profile

> **Important**
>
> Lorsque vous supprimez le mot de passe d'un utilisateur, ce dernier ne peut plus se connecter à l'interface Console de gestion AWS. Si l'utilisateur dispose de clés d'accès actives, celles-ci continuent de fonctionner et autorisent l'accès via les appels de fonction AWS CLI PowerShell, Outils pour Windows ou AWS API. Lorsque vous utilisez les AWS CLI Outils pour Windows PowerShell ou l' AWS API pour supprimer un utilisateur de votre compte Compte AWS, vous devez d'abord supprimer le mot de passe à l'aide de cette opération. Pour de plus amples informations, veuillez consulter Suppression d'un utilisateur IAM (AWS CLI).

Pour révoquer les sessions de console actives d'un utilisateur avant une heure spécifiée (AWS CLI)

1. Pour intégrer une politique en ligne qui révoque les sessions de console actives d'un utilisateur IAM avant une heure spécifiée, utilisez la politique en ligne suivante et exécutez cette commande : aws iam put-user-policy

   Cette politique en ligne récuse toutes les autorisations et inclut la clé de condition `aws:TokenIssueTime`. Il révoque les sessions de console actives de l'utilisateur avant l'heure spécifiée dans l'élément `Condition` de la politique en ligne. Remplacez la valeur de la clé de condition `aws:TokenIssueTime` par votre propre valeur.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {
        "aws:TokenIssueTime": "2014-05-07T23:47:00Z"
      }
    }
  }
}
```

2. (Facultatif) Pour répertorier les noms des politiques en ligne intégrées à l'utilisateur IAM, exécutez cette commande : aws iam list-user-policies
3. (Facultatif) Pour afficher la politique en ligne nommée intégrée à l'utilisateur IAM, exécutez cette commande : aws iam get-user-policy

Création, modification ou suppression d'un mot de passe utilisateur IAM (AWS API)

Vous pouvez utiliser l' AWS API pour gérer les mots de passe de vos utilisateurs IAM.

Pour créer un mot de passe (AWS API)

1. (Facultatif) Pour déterminer si un utilisateur possède un mot de passe, appelez cette opération : GetLoginProfile
2. Pour créer un mot de passe, appelez cette opération : CreateLoginProfile

Pour modifier le mot de passe d'un utilisateur (AWS API)

1. (Facultatif) Pour déterminer si un utilisateur possède un mot de passe, appelez cette opération : GetLoginProfile
2. Pour modifier un mot de passe, appelez cette opération : UpdateLoginProfile

Pour supprimer (désactiver) le mot de passe d'un utilisateur (AWS API)

1. (Facultatif) Pour déterminer si un utilisateur possède un mot de passe, exécutez cette commande : GetLoginProfile
2. (Facultatif) Pour déterminer la date de dernière utilisation d'un mot de passe, exécutez cette commande : GetUser
3. Pour supprimer un mot de passe, exécutez cette commande : DeleteLoginProfile

> **Important**
>
> Lorsque vous supprimez le mot de passe d'un utilisateur, ce dernier ne peut plus se connecter à l'interface Console de gestion AWS. Si l'utilisateur dispose de clés d'accès actives, celles-ci continuent de fonctionner et autorisent l'accès via les appels de fonction AWS CLI PowerShell, Outils pour Windows ou AWS API. Lorsque vous utilisez les AWS CLI Outils pour Windows PowerShell ou l' AWS API pour supprimer un utilisateur de votre compte Compte AWS, vous devez d'abord supprimer le mot de passe à l'aide de cette opération. Pour de plus amples informations, veuillez consulter Suppression d'un utilisateur IAM (AWS CLI).

Pour révoquer les sessions de console actives d'un utilisateur avant une heure spécifiée (AWS API)

1. Pour intégrer une politique en ligne qui révoque les sessions de console actives d'un utilisateur IAM avant une heure spécifiée, utilisez la stratégie en ligne suivante et exécutez cette commande : PutUserPolicy

   Cette politique en ligne récuse toutes les autorisations et inclut la clé de condition `aws : TokenIssueTime`. Il révoque les sessions de console actives de l'utilisateur avant l'heure spécifiée dans l'élément `Condition` de la politique en ligne. Remplacez la valeur de la clé de condition `aws:TokenIssueTime` par votre propre valeur.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Deny",
```

```
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {
        "aws:TokenIssueTime": "2014-05-07T23:47:00Z"
      }
    }
  }
}
```

2. (Facultatif) Pour répertorier les noms des politiques en ligne intégrées à l'utilisateur IAM, exécutez cette commande : ListUserPolicies
3. (Facultatif) Pour afficher la politique en ligne nommée intégrée à l'utilisateur IAM, exécutez cette commande : GetUserPolicy

Octroi aux utilisateurs IAM de l'autorisation de modification de leurs propres mots de passe

> **Note**
>
> Les utilisateurs dotés d'identités fédérées utiliseront le processus défini par leur fournisseur d'identité pour modifier leurs mots de passe. Il est recommandé d'obliger les utilisateurs humains à utiliser la fédération avec un fournisseur d'identité pour accéder à l' AWS aide d'informations d'identification temporaires.

Vous pouvez accorder aux utilisateurs IAM l'autorisation de modifier leurs propres mots de passe lors de la connexion à la Console de gestion AWS. Vous pouvez effectuer cette opération de deux manières :

- Autorisez tous les utilisateurs IAM du compte à modifier leurs propres mots de passe.
- Autorisez uniquement les utilisateurs IAM sélectionnés à modifier leurs propres mots de passe. Dans ce scénario, vous désactivez l'option permettant à tous les utilisateurs de modifier leurs propres mots de passe et vous utilisez une politique IAM pour accorder des autorisations uniquement à certains utilisateurs. Cette approche permet à ces utilisateurs de modifier leurs propres mots de passe et éventuellement d'autres informations d'identification telles que leurs propres clés d'accès.

> **Important**
>
> Nous vous recommandons de définir une politique de mot de passe personnalisée qui oblige les utilisateurs IAM à créer des mots de passe d'un niveau de sécurité élevé.

Permettre à tous les utilisateurs IAM de modifier leur propre mot de passe

Console

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation, choisissez Paramètres du compte.
3. Dans la section Password policy (Politique de mot de passe), sélectionnez Edit (Modifier).
4. Choisissez Custom (Personnalisé) pour utiliser une politique de mot de passe personnalisée.
5. Sélectionnez Allow users to change their own password (Autoriser les utilisateurs à modifier leur propre mot de passe), puis cliquez sur Save changes (Enregistrer les modifications). Cela permet à tous les utilisateurs du compte d'accéder à l'action `iam:ChangePassword` uniquement pour leur propre utilisateur et à l'action `iam:GetAccountPasswordPolicy`.
6. Fournissez aux utilisateurs les instructions suivantes pour modifier leurs mots de passe : Modification par l'utilisateur IAM de son propre mot de passe.

AWS CLI

Exécutez la commande suivante :

- `aws iam update-account-password-policy`

API

Pour mettre à jour la politique de mot de passe du compte, effectuez l'opération suivante :

- `UpdateAccountPasswordPolicy`

Pour autoriser les utilisateurs IAM sélectionnés à modifier leur propre mot de passe

Console

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation, choisissez Paramètres du compte.
3. Dans la section Paramètres du compte, vérifiez que l'option Autoriser les utilisateurs à modifier leur propre mot de passe n'est pas sélectionnée. Si cette case à cocher est activée, tous les utilisateurs peuvent modifier leurs propres mots de passe. (Reportez-vous à la procédure précédente.)
4. Créez les utilisateurs qui devraient être autorisés à modifier leurs propres mots de passe, s'il n'existe pas encore. Pour en savoir plus, consultez Créez un utilisateur IAM dans votre Compte AWS.
5. (Facultatif) Créez un groupe IAM pour les utilisateurs qui doivent être autorisés à modifier leurs mots de passe, puis ajoutez les utilisateurs de l'étape précédente au groupe. Pour en savoir plus, consultez Groupes d'utilisateurs IAM.
6. Affectez la politique suivante au groupe. Pour de plus amples informations, veuillez consulter Gestion des politiques IAM.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:GetAccountPasswordPolicy",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:ChangePassword",
      "Resource": "arn:aws:iam::*:user/${aws:username}"
    }
  ]
}
```

Cette politique donne accès à l'ChangePasswordaction, qui permet aux utilisateurs de modifier uniquement leurs propres mots de passe depuis la console AWS CLI, les Outils pour Windows PowerShell ou l'API. Elle donne également accès à l'GetAccountPasswordPolicyaction, qui permet à l'utilisateur de consulter la politique de mot de passe actuelle ; cette autorisation est requise pour que l'utilisateur puisse consulter la politique de mot de passe du compte sur la page Modifier le mot de passe. L'utilisateur doit être autorisé à lire la politique de mot de passe actuelle pour s'assurer que le mot de passe modifié répond aux exigences de cette politique.

7. Fournissez aux utilisateurs les instructions suivantes pour modifier leurs mots de passe : Modification par l'utilisateur IAM de son propre mot de passe.

Pour plus d’informations

Pour plus d'informations sur la gestion des informations d'identification, consultez les rubriques suivantes :

- Octroi aux utilisateurs IAM de l’autorisation de modification de leurs propres mots de passe
- Mots de passe utilisateur dans AWS
- Définition d’une politique de mot de passe du compte pour les utilisateurs IAM
- Gestion des politiques IAM
- Modification par l'utilisateur IAM de son propre mot de passe

Modification par l'utilisateur IAM de son propre mot de passe

Si vous avez été autorisé à modifier votre propre mot de passe utilisateur IAM, vous pouvez utiliser une page spéciale Console de gestion AWS à cet effet. Vous pouvez également utiliser l' AWS API AWS CLI or.

Rubriques

- Autorisations nécessaires
- Comment les utilisateurs IAM modifient leur mot de passe (console)
- Comment les utilisateurs IAM modifient leur propre mot de passe (AWS CLI or AWS API)

Autorisations nécessaires

Pour modifier le mot de passe pour votre propre utilisateur IAM, vous devez disposer des autorisations de la politique suivante : AWS : autorise les utilisateurs IAM à modifier leur propre mot de passe pour la console sur la page Informations d'identification de sécurité.

Comment les utilisateurs IAM modifient leur mot de passe (console)

La procédure suivante décrit comment les utilisateurs IAM peuvent utiliser le Console de gestion AWS pour modifier leur propre mot de passe.

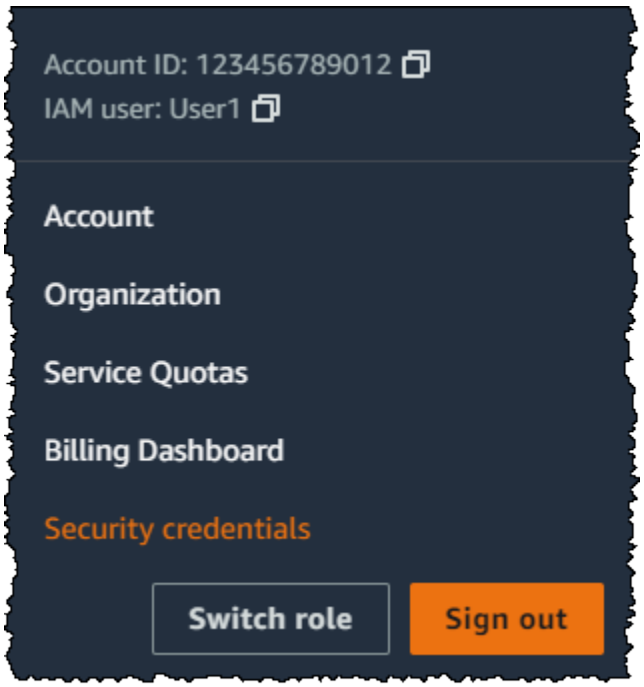
Pour modifier votre propre mot de passe utilisateur IAM (console)

1. Utilisez votre AWS identifiant ou alias de compte, votre nom d'utilisateur IAM et votre mot de passe pour vous connecter à la console IAM.

   > **Note**
   >
   > Pour votre commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser votre nom d'utilisateur IAM et les informations de votre compte. Si vous vous êtes déjà connecté en tant qu'utilisateur différent, sélectionnez Sign in to a different account (Se connecter à un compte différent) en bas de la page pour revenir à la page de connexion principale. À partir de là, vous pouvez saisir votre identifiant de AWS compte ou votre alias de compte pour être redirigé vers la page de connexion utilisateur IAM de votre compte.

   Pour obtenir votre Compte AWS identifiant, contactez votre administrateur.

2. Dans la barre de navigation, en haut à droite, choisissez votre nom d'utilisateur, puis Security credentials (Informations d'identification de sécurité).



3. Dans l'onglet Informations d'identification AWS IAM, sélectionnez Mettre à jour le mot de passe.
4. Dans le champ Current password (Mot de passe actuel), saisissez le mot de passe que vous utilisez actuellement. Entrez un nouveau mot de passe dans le champ New password (Nouveau mot de passe), puis confirmez-le dans le champ Confirm new password (Confirmer le nouveau mot de passe). Ensuite, choisissez l'option Mettre à jour le mot de passe.

> **Note**
>
> Le nouveau mot de passe doit répondre aux exigences de la politique de mot de passe du compte. Pour de plus amples informations, veuillez consulter Définition d'une politique de mot de passe du compte pour les utilisateurs IAM.

Comment les utilisateurs IAM modifient leur propre mot de passe (AWS CLI or AWS API)

La procédure suivante décrit comment les utilisateurs IAM peuvent utiliser l' AWS API AWS CLI or pour modifier leur propre mot de passe.

Pour modifier votre mot de passe IAM, utilisez ce qui suit :

- AWS CLI: `aws iam change-password`
- AWS API : `ChangePassword`

Gestion des clés d'accès pour les utilisateurs IAM

> **Important**
>
> Les bonnes pratiques consistent à utiliser des informations d'identification de sécurité temporaires (comme des rôles IAM) plutôt que de créer des informations d'identification à long terme comme des clés d'accès. Avant de créer des clés d'accès, passez en revue les alternatives aux clés d'accès à long terme.

Les clés d'accès sont les informations d'identification à long terme d'un utilisateur IAM ou du Utilisateur racine d'un compte AWS. Vous pouvez utiliser les clés d'accès pour signer des demandes programmatiques adressées à l' AWS API AWS CLI or (directement ou à l'aide du AWS SDK). Pour de plus amples informations, veuillez consulter Accès programmatique avec AWS Informations d'identification de sécurité.

Les clés d'accès se composent de deux parties : un ID de clé d'accès (par exemple, `AKIAIOSFODNN7EXAMPLE`) et une clé d'accès secrète (par exemple, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Vous devez utiliser à la fois l'ID de la clé d'accès et la clé d'accès secrète pour authentifier vos demandes.

Lorsque vous créez une paire de clé d'accès, enregistrez l'ID de clé d'accès et la clé d'accès secrète dans un emplacement sécurisé. La clé d'accès secrète ne peut être récupérée qu'au moment de sa création. Si vous perdez votre clé d'accès secrète, vous devez supprimer la clé d'accès et en créer une nouvelle. Pour plus d'informations, consultez Mise à jour des clés d'accès.

Vous pouvez avoir un maximum de deux clés d'accès par utilisateur.

> **Important**
>
> Les utilisateurs IAM dotés de clés d'accès constituent un risque pour la sécurité des comptes. Gérez vos clés d'accès en toute sécurité. Ne communiquez pas vos clés d'accès à des tiers non autorisés, même pour vous aider à trouver les identifiants de votre compte. Ce faisant, vous pouvez donner à quelqu'un un accès permanent à votre compte.
> Lorsque vous utilisez des clés d'accès, soyez conscient de ce qui suit :
>
> - N'utilisez PAS les informations d'identification racine de votre compte pour créer des clés d'accès.

- N'incluez PAS de clés d'accès ou d'informations d'identification dans vos fichiers d'application.
- N'incluez PAS de fichiers contenant des clés d'accès ou des informations d'identification dans votre zone de projet.
- Les clés d'accès ou les informations d'identification stockées dans le fichier AWS d'informations d'identification partagé sont stockées en texte brut.

Recommandations de surveillance

Après avoir créé les clés d'accès :

- AWS CloudTrail À utiliser pour surveiller l'utilisation des clés d'accès et détecter toute tentative d'accès non autorisée. Pour de plus amples informations, veuillez consulter Journalisation IAM et AWS STS Appels d'API avec AWS CloudTrail.
- Configurez des CloudWatch alarmes pour avertir les administrateurs en cas de tentative de refus d'accès afin de détecter les activités malveillantes. Pour plus d'informations, consultez le guide de CloudWatch l'utilisateur Amazon.
- Vérifiez, mettez à jour et supprimez régulièrement les clés d'accès selon les besoins.

Les rubriques suivantes détaillent les tâches de gestion associées aux clés d'accès.

Rubriques

- Contrôler l'utilisation des clés d'accès en associant une politique en ligne à un utilisateur IAM
- Autorisations requises pour gérer les clés d'accès
- Gestion par les utilisateurs IAM de leurs propres clés d'accès
- Gestion des clés d'accès d'un utilisateur IAM par un administrateur IAM
- Mise à jour des clés d'accès
- Sécurisation des clés d'accès

Contrôler l'utilisation des clés d'accès en associant une politique en ligne à un utilisateur IAM

Nous recommandons, à titre de bonne pratique, que les charges de travail utilisent des informations d'identification temporaires avec des rôles IAM pour accéder à AWS. Les utilisateurs IAM disposant

de clés d'accès doivent se voir attribuer un accès de moindre privilège et bénéficier d'une authentification multifactorielle (MFA). Pour plus d'informations sur l'endossement des rôles IAM, consultez Méthodes pour assumer un rôle.

Toutefois, si vous créez un test de validation de principe d'une automatisation de service ou d'un autre cas d'utilisation à court terme, et que vous choisissez d'exécuter des charges de travail à l'aide d'un utilisateur IAM doté de clés d'accès, nous vous recommandons d'utiliser des conditions de politique pour restreindre davantage l'accès à ses informations d'identification utilisateur IAM.

Dans cette situation, vous pouvez soit créer une politique à durée limitée qui fait expirer les informations d'identification après le délai spécifié, soit, si vous exécutez une charge de travail à partir d'un réseau sécurisé, utiliser une politique de restriction IP.

Pour ces deux cas d'utilisation, vous pouvez utiliser une politique en ligne associée à l'utilisateur IAM qui dispose des clés d'accès.

Pour configurer une politique limitée dans le temps pour un utilisateur IAM

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le volet de navigation, choisissez Utilisateurs, puis sélectionnez l'utilisateur pour le cas d'utilisation à court terme. Si vous n'avez pas encore créé l'utilisateur, vous pouvez le créer maintenant.
3. Sur la page Détails de l'utilisateur, choisissez l'onglet Autorisations.
4. Sélectionnez Ajouter des autorisations, puis Créer la politique en ligne.
5. Dans la section Éditeur de politiques, sélectionnez JSON pour afficher l'éditeur JSON.
6. Dans l'éditeur JSON, saisissez la politique suivante en remplaçant la valeur de l'horodatage `aws:CurrentTime` par la date et l'heure d'expiration souhaitées :

   JSON

```
{
"Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
```

```
      "Condition": {
      "DateGreaterThan": {
      "aws:CurrentTime": "2025-03-01T00:12:00Z"
        }
      }
    }
  ]
}
```

Cette politique utilise l'effet `Deny` pour restreindre toutes les actions sur toutes les ressources après la date spécifiée. La condition `DateGreaterThan` compare l'heure actuelle avec l'horodatage que vous avez défini.

7. Sélectionnez Suivant pour accéder à la page Vérifier et créer. Dans Détails de la politique, sous Nom de la politique, saisissez un nom pour la politique, puis sélectionnez Créer une politique.

Une fois la politique créée, elle s'affiche dans l'onglet Autorisations de l'utilisateur. Lorsque l'heure actuelle est supérieure ou égale à l'heure spécifiée dans la politique, l'utilisateur n'aura plus accès aux AWS ressources. Veillez à informer les développeurs de charge de travail de la date d'expiration que vous avez spécifiée pour ces clés d'accès.

Pour configurer une politique de restriction IP pour un utilisateur IAM

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le volet de navigation, sélectionnez Utilisateurs, puis sélectionnez l'utilisateur qui exécutera la charge de travail à partir du réseau sécurisé. Si vous n'avez pas encore créé l'utilisateur, vous pouvez le créer maintenant.
3. Sur la page Détails de l'utilisateur, choisissez l'onglet Autorisations.
4. Sélectionnez Ajouter des autorisations, puis Créer la politique en ligne.
5. Dans la section Éditeur de politiques, sélectionnez JSON pour afficher l'éditeur JSON.
6. Copiez la politique IAM suivante dans l'éditeur JSON et modifiez les adresses IPv4 ou IPv6 publiques, ou les plages, selon vos besoins. Vous pouvez l'utiliser https://checkip.amazonaws.com/ pour déterminer votre adresse IP publique actuelle. Vous pouvez spécifier des adresses IP individuelles ou des plages d'adresses IP à l'aide de la notation oblique. Pour de plus amples informations, veuillez consulter aws : SourceIp.

> **Note**
>
> Les adresses IP ne doivent pas être masquées par un VPN ou un serveur proxy.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid":"IpRestrictionIAMPolicyForIAMUser",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "NotIpAddress": {
          "aws:SourceIp": [
            "203.0.113.0/24",
            "2001:DB8:1234:5678::/64",
            "203.0.114.1"
          ]
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
}
```

Cet exemple de politique refuse l'utilisation des clés d'accès d'un utilisateur IAM lorsque cette politique est appliquée, sauf si la demande provient des réseaux (spécifiés en notation CIDR) « 203.0.113 ». 0/24», « 2001 DB8:1234:5678 : :/64 », ou l'adresse IP spécifique « 203.0.114.1 »

7. Sélectionnez Suivant pour accéder à la page Vérifier et créer. Dans Détails de la politique, sous Nom de la politique, saisissez un nom pour la politique, puis sélectionnez Créer une politique.

Une fois la politique créée, elle s’affiche dans l’onglet Autorisations de l’utilisateur.

Vous pouvez également appliquer cette politique en tant que politique de contrôle des services (SCP) à plusieurs AWS comptes dans AWS Organizations, nous vous recommandons d'utiliser une condition supplémentaire, `aws:PrincipalArn` pour que cette déclaration de politique ne s'applique qu'aux utilisateurs IAM des AWS comptes soumis à ce SCP. La politique suivante inclut cette mise à jour :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "IpRestrictionServiceControlPolicyForIAMUsers",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "NotIpAddress": {
          "aws:SourceIp": [
            "203.0.113.0/24",
            "2001:DB8:1234:5678::/64",
            "203.0.114.1"
          ]
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        },
        "ArnLike": {
          "aws:PrincipalArn": "arn:aws:iam::*:user/*"
        }
      }
    }
  ]
}
```

Autorisations requises pour gérer les clés d'accès

> **Note**
>
> `iam:TagUser` est une autorisation facultative permettant d'ajouter et de modifier des descriptions pour la clé d'accès. Pour de plus amples informations, consultez Balisage des utilisateurs IAM.

Pour créer des clés d'accès pour votre propre utilisateur IAM, vous devez disposer des autorisations de la politique suivante :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Pour mettre à jour des clés d'accès pour votre propre utilisateur IAM, vous devez disposer des autorisations de la politique suivante :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "ManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:GetAccessKeyLastUsed",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Gestion par les utilisateurs IAM de leurs propres clés d'accès

Les administrateurs IAM peuvent accorder aux utilisateurs IAM l'autorisation de gérer eux-mêmes leurs clés d'accès en attachant la politique décrite dans Autorisations requises pour gérer les clés d'accès.

Avec ces autorisations, l'utilisateur IAM peut utiliser les procédures suivantes pour créer, activer, désactiver et supprimer les clés d'accès associées à son nom d'utilisateur.

Rubriques

- Création d'une clé d'accès pour vous-même (console)
- Désactivation de votre clé d'accès (console)
- Activation de votre clé d'accès (console)
- Suppression de votre clé d'accès (console)

Création d'une clé d'accès pour vous-même (console)

Si les autorisations appropriées vous ont été accordées, vous pouvez les utiliser Console de gestion AWS pour créer des clés d'accès pour vous-même.

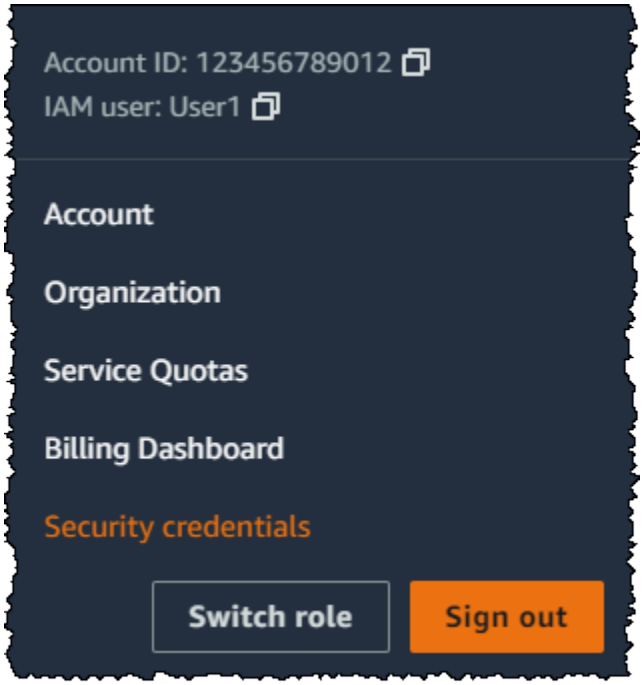
Pour créer vos propres clés d'accès (console)

1. Utilisez votre AWS identifiant ou alias de compte, votre nom d'utilisateur IAM et votre mot de passe pour vous connecter à la console IAM.

   > **Note**
   >
   > Pour votre commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser votre nom d'utilisateur IAM et les informations de votre compte. Si vous vous êtes déjà connecté en tant qu'utilisateur différent, sélectionnez Sign in to a different account (Se connecter à un compte différent) en bas de la page pour revenir à la page de connexion principale. À partir de là, vous pouvez saisir votre identifiant de AWS compte ou votre alias de compte pour être redirigé vers la page de connexion utilisateur IAM de votre compte.

   Pour obtenir votre Compte AWS identifiant, contactez votre administrateur.

2. Dans la barre de navigation, en haut à droite, choisissez votre nom d'utilisateur, puis Security credentials (Informations d'identification de sécurité).



3. Dans la section Clés d'accès, choisissez Créer une clé d'accès. Si vous avez déjà deux clés d'accès, ce bouton est désactivé et vous devez supprimer une clé d'accès avant de pouvoir en créer une nouvelle.

4. Sur la page des bonnes pratiques et alternatives en matière de clés d'accès, choisissez votre cas d'utilisation pour découvrir les options supplémentaires qui peuvent vous aider à éviter de créer une clé d'accès à long terme. Si vous déterminez que votre cas d'utilisation nécessite toujours une clé d'accès, choisissez Other (Autre), puis Next (Suivant).
5. (Facultatif) Définissez une valeur de balise de description pour la clé d'accès. Cela permet d'ajouter une paire clé-valeur de balise à votre utilisateur IAM. Cela peut vous aider à identifier et à mettre à jour les clés d'accès par la suite. La clé de balise est définie sur l'identifiant de la clé d'accès. La valeur de balise est définie selon la description de la clé d'accès que vous spécifiez. Lorsque vous avez terminé, sélectionnez Create access key (Créer la clé d'accès).
6. Sur la page Retrieve access keys (Récupérer les clés d'accès), choisissez Show (Afficher) (pour révéler la valeur de la clé d'accès secrète de votre utilisateur) ou Download .csv file (Télécharger le fichier .csv). C'est le seul moment où vous pouvez enregistrer votre clé d'accès secrète. Après avoir enregistré votre clé d'accès secrète dans un emplacement sécurisé, sélectionnez Done (Terminé).

Désactivation de votre clé d'accès (console)

Si les autorisations appropriées vous ont été accordées, vous pouvez les utiliser Console de gestion AWS pour désactiver votre clé d'accès.

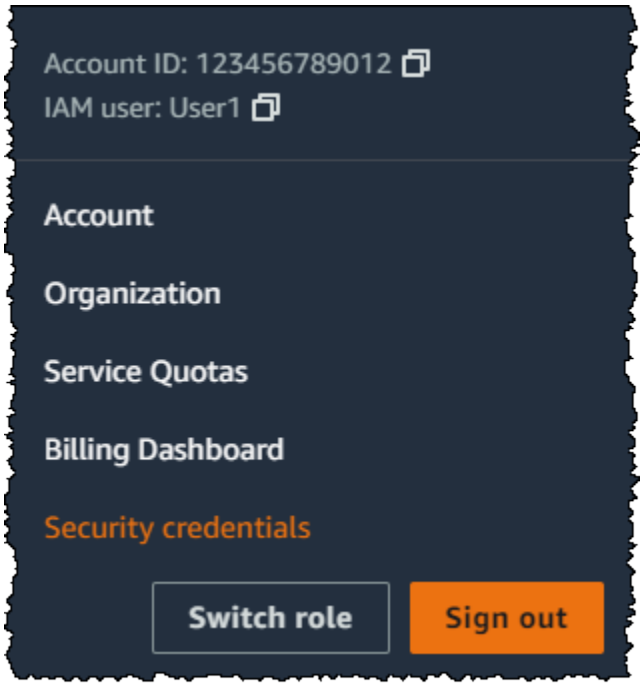
Pour désactiver une clé d'accès

1. Utilisez votre AWS identifiant ou alias de compte, votre nom d'utilisateur IAM et votre mot de passe pour vous connecter à la console IAM.

   > **Note**
   >
   > Pour votre commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser votre nom d'utilisateur IAM et les informations de votre compte. Si vous vous êtes déjà connecté en tant qu'utilisateur différent, sélectionnez Sign in to a different account (Se connecter à un compte différent) en bas de la page pour revenir à la page de connexion principale. À partir de là, vous pouvez saisir votre identifiant de AWS compte ou votre alias de compte pour être redirigé vers la page de connexion utilisateur IAM de votre compte.

   Pour obtenir votre Compte AWS identifiant, contactez votre administrateur.

2. Dans la barre de navigation, en haut à droite, choisissez votre nom d'utilisateur, puis Security credentials (Informations d'identification de sécurité).



3. Dans la section Access keys (Clés d'accès), recherchez la clé que vous souhaitez désactiver, puis choisissez Actions, puis Deactivate (Désactiver). À l'invite de confirmation, cliquez sur Deactivate (Désactiver). Une clé d'accès désactivée compte toujours dans votre limite de deux clés d'accès.

Activation de votre clé d'accès (console)

Si vous avez obtenu les autorisations appropriées, vous pouvez utiliser le Console de gestion AWS pour activer votre clé d'accès.

Pour activer une clé d'accès

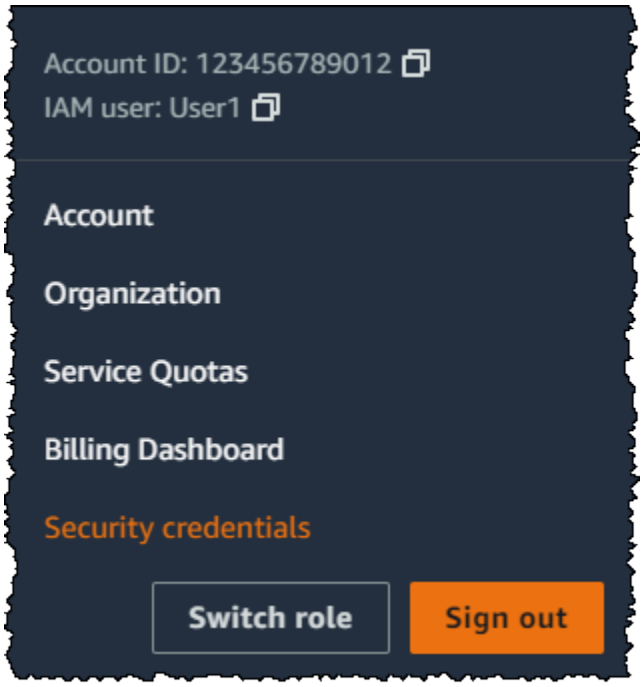
1. Utilisez votre AWS identifiant ou alias de compte, votre nom d'utilisateur IAM et votre mot de passe pour vous connecter à la console IAM.

> **Note**
>
> Pour votre commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser votre nom d'utilisateur IAM et les informations de votre compte. Si vous vous êtes déjà connecté en tant qu'utilisateur différent, sélectionnez Sign in to a different account (Se connecter à un compte différent) en bas de la page pour revenir à la page

de connexion principale. À partir de là, vous pouvez saisir votre identifiant de AWS compte ou votre alias de compte pour être redirigé vers la page de connexion utilisateur IAM de votre compte.

Pour obtenir votre Compte AWS identifiant, contactez votre administrateur.

2. Dans la barre de navigation, en haut à droite, choisissez votre nom d'utilisateur, puis Security credentials (Informations d'identification de sécurité).



3. Dans la section Access keys (Clés d'accès), recherchez la clé que vous souhaitez activer, puis choisissez Actions, puis Activate (Activer).

Suppression de votre clé d'accès (console)

Si les autorisations appropriées vous ont été accordées, vous pouvez les utiliser Console de gestion AWS pour supprimer votre clé d'accès.

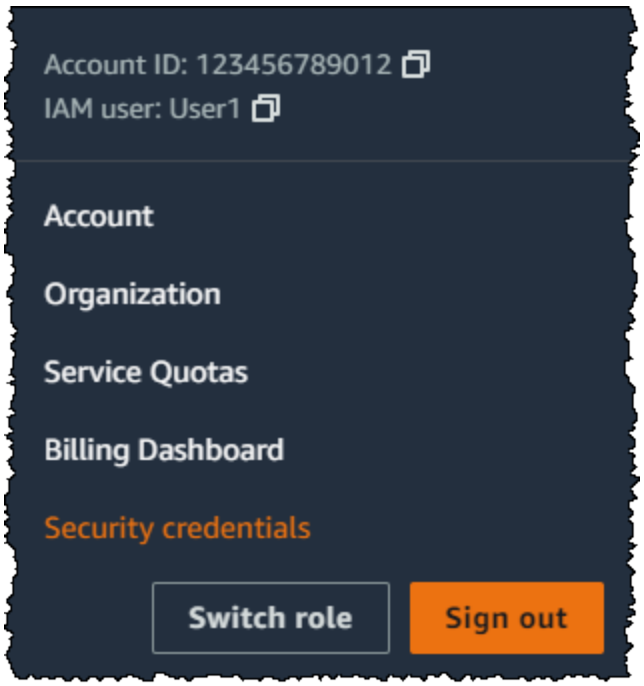
Pour supprimer une clé d'accès dont vous n'avez plus besoin

1. Utilisez votre AWS identifiant ou alias de compte, votre nom d'utilisateur IAM et votre mot de passe pour vous connecter à la console IAM.

> **Note**
>
> Pour votre commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser votre nom d'utilisateur IAM et les informations de votre compte. Si vous vous êtes déjà connecté en tant qu'utilisateur différent, sélectionnez Sign in to a different account (Se connecter à un compte différent) en bas de la page pour revenir à la page de connexion principale. À partir de là, vous pouvez saisir votre identifiant de AWS compte ou votre alias de compte pour être redirigé vers la page de connexion utilisateur IAM de votre compte.

Pour obtenir votre Compte AWS identifiant, contactez votre administrateur.

2. Dans la barre de navigation, en haut à droite, choisissez votre nom d'utilisateur, puis Security credentials (Informations d'identification de sécurité).



3. Dans la section Access keys (Clés d'accès), recherchez la clé que vous souhaitez suprimer, puis choisissez Actions, puis Delete (Supprimer). Suivez les instructions de la boîte de dialogue pour tout d'abord désactiver, puis confirmer la suppression. Nous vous recommandons de vérifier que la clé d'accès n'est plus utilisée avant de la supprimer définitivement.

Gestion des clés d’accès d’un utilisateur IAM par un administrateur IAM

Les administrateurs IAM peuvent créer, activer, désactiver et supprimer les clés d’accès associées à des utilisateurs IAM individuels. Ils peuvent également répertorier les utilisateurs IAM du compte qui possèdent des clés d’accès et localiser quel utilisateur IAM possède une clé d’accès spécifique.

Rubriques

- Pour créer une clé d’accès pour un utilisateur IAM
- Pour désactiver une clé d’accès pour un utilisateur IAM
- Pour activer une clé d’accès pour un utilisateur IAM
- Pour supprimer une clé d’accès d’un utilisateur IAM
- Pour lister les clés d’accès d’un utilisateur IAM
- Pour afficher tous les ID de clé d’accès des utilisateurs de votre compte
- Pour rechercher un utilisateur à l’aide d’un ID de clé d’accès
- Pour recherche l’utilisation la plus récente d’un ID de clé d’accès

Pour créer une clé d’accès pour un utilisateur IAM

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom d’utilisateur pour accéder à la page des détails de l’utilisateur.
4. Dans l’onglet Informations d’identification, dans la section Clés d’accès, sélectionnez Créer une clé d’accès.

   Si le bouton est désactivé, vous devez supprimer l'une des clés existantes avant de pouvoir en créer une nouvelle.
5. Sur la page Bonnes pratiques et alternatives en matière de clés d'accès, passez en revue les bonnes pratiques et alternatives. Choisissez votre cas d'utilisation pour découvrir les options supplémentaires qui peuvent vous aider à éviter de créer une clé d'accès à long terme.
6. Si vous déterminez que votre cas d'utilisation nécessite toujours une clé d'accès, choisissez Other (Autre), puis Next (Suivant).

7. (Facultatif) Sur la page Définir une balise de description, vous pouvez ajouter une balise de description à la clé d'accès pour faciliter le suivi de votre clé d'accès. Sélectionnez Créer une clé d'accès.
8. Sur la page Retrieve access key page (Récupérer les clés d'accès), choisissez Show (Afficher) pour révéler la valeur de la clé d'accès secrète de votre utilisateur.
9. Pour enregistrer l'ID de clé d'accès et la clé d'accès secrète dans un fichier `.csv`, dans un emplacement sécurisé sur votre ordinateur, sélectionnez le bouton Download .csv file (Télécharger un fichier .csv).

> **Important**
>
> Ce sera votre seule occasion de consulter ou télécharger la clé d'accès nouvellement créée et vous ne pourrez pas la récupérer. Assurez-vous de conserver votre clé d'accès en sécurité.

Lorsque vous créez une clé d'accès pour votre utilisateur, cette paire de clés est activée par défaut et votre utilisateur peut immédiatement l'utiliser.

AWS CLI

Exécutez la commande suivante :

- `aws iam create-access-key`

API

Appelez l'opération suivante :

- `CreateAccessKey`

Pour désactiver une clé d'accès pour un utilisateur IAM

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez utilisateurs.

3. Choisissez le nom d'utilisateur pour accéder à la page des détails de l'utilisateur.
4. Dans l'onglet Informations d'identification de sécurité, dans la section Clés d'accès, choisissez le menu déroulant Actions, puis sélectionnez Désactiver.
5. Dans la boîte de dialogue Désactiver, confirmez que vous souhaitez désactiver la clé d'accès en sélectionnant Désactiver

Une fois qu'une clé d'accès est désactivée, elle ne peut plus être utilisée par les appels d'API. Vous pouvez la réactiver si nécessaire.

AWS CLI

Exécutez la commande suivante :

- `aws iam update-access-key`

API

Appelez l'opération suivante :

- `UpdateAccessKey`

Pour activer une clé d'accès pour un utilisateur IAM

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom d'utilisateur pour accéder à la page des détails de l'utilisateur.
4. Dans l'onglet Informations d'identification de sécurité, dans la section Clés d'accès, choisissez le menu déroulant Actions, puis sélectionnez Activer.

Une fois qu'une clé d'accès est activée, elle peut être utilisée par les appels d'API. Vous pouvez la désactiver à nouveau si nécessaire.

AWS CLI

Exécutez la commande suivante :

- `aws iam update-access-key`

API

Appelez l'opération suivante :

- `UpdateAccessKey`

Pour supprimer une clé d'accès d'un utilisateur IAM

Une fois qu'une clé d'accès a été désactivée, supprimez-la si elle n'est plus nécessaire.

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom d'utilisateur pour accéder à la page des détails de l'utilisateur.
4. Dans l'onglet Informations d'identification de sécurité, dans la section Clés d'accès, choisissez le menu déroulant Actions pour la clé d'accès inactive, puis sélectionnez Supprimer.
5. Dans la boîte de dialogue Supprimer, confirmez que vous souhaitez supprimer la clé d'accès en saisissant son ID dans le champ de saisie de texte, puis en sélectionnant Supprimer.

Une fois que la clé d'accès est supprimée, elle ne peut pas être récupérée.

AWS CLI

Exécutez la commande suivante :

- `aws iam delete-access-key`

API

Appelez l'opération suivante :

- `DeleteAccessKey`

Pour lister les clés d'accès d'un utilisateur IAM

Vous pouvez consulter la liste des ID de clé d'accès associés à un utilisateur IAM.

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom d'utilisateur pour accéder à la page des détails de l'utilisateur.
4. Dans l'onglet Informations d'identification de sécurité, la section Clés d'accès répertorie les ID de clé d'accès de l'utilisateur, y compris le statut de chaque clé affichée.

   > **Note**
   >
   > Seul l'ID de clé d'accès de l'utilisateur est visible. La clé d'accès secrète peut être récupérée uniquement au moment de la création de la clé.

   Chaque utilisateur IAM peut avoir deux clés d'accès.

AWS CLI

Exécutez la commande suivante :

- `aws iam list-access-keys`

API

Appelez l'opération suivante :

- `ListAccessKeys`

Pour afficher tous les ID de clé d'accès des utilisateurs de votre compte

Vous pouvez afficher une liste des ID de clé d'accès des utilisateurs de votre Compte AWS.

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom d'utilisateur pour accéder à la page des détails de l'utilisateur.
4. Au besoin, ajoutez la colonne ID de clé d'accès à la table des utilisateurs en procédant comme suit :

   a. Au-dessus de la table à l'extrême droite, choisissez l'icône des Préférences ().

   b. Dans la boîte de dialogue Préférences, sous Sélectionner les colonnes visibles, activez l'ID de clé d'accès.

   c. Choisissez Confirmer pour revenir à la liste des utilisateurs. La liste est mise à jour pour inclure l'ID de clé d'accès.

5. La colonne ID de clé d'accès indique l'état de chaque clé d'accès, suivi de son ID ; par exemple, **`Active - AKIAIOSFODNN7EXAMPLE`** ou **`Inactive - AKIAI44QH8DHBEXAMPLE`**.

   Vous pouvez utiliser ces informations pour afficher et copier les ID de clé d'accès des utilisateurs ayant une ou deux clés d'accès. La colonne affiche **`-`** pour les utilisateurs ne disposant pas d'une clé d'accès.

   > **Note**
   >
   > La clé d'accès secrète peut être récupérée uniquement au moment de la création de la clé.

   Chaque utilisateur IAM peut avoir deux clés d'accès.

Pour rechercher un utilisateur à l'aide d'un ID de clé d'accès

Vous pouvez utiliser un ID de clé d'accès pour rechercher un utilisateur dans votre Compte AWS.

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le volet de navigation, dans la zone de recherche, saisissez l'ID de clé d'accès, par exemple AKIAI44QH8DHBEXAMPLE.
3. L'utilisateur IAM auquel l'ID de clé d'accès est associé apparaît dans le volet de navigation. Choisissez le nom d'utilisateur pour accéder à la page des détails de l'utilisateur.

Pour recherche l'utilisation la plus récente d'un ID de clé d'accès

L'utilisation la plus récente d'une clé d'accès est affichée dans la liste des utilisateurs sur la page des utilisateurs IAM, sur la page détaillée de l'utilisateur, et fait partie du rapport d'informations d'identification.

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans la liste des utilisateurs, consultez la colonne Dernière utilisation de clé d'accès.

   Si la colonne n'est pas affichée, cliquez sur l'icône Préférences
   ()
   et sous Sélectionner les colonnes visibles, activez Dernière utilisation de clé d'accès pour afficher la colonne.
3. (facultatif) Dans le volet de navigation, sous Rapports d'accès, sélectionnez Rapport d'informations d'identification pour télécharger un rapport qui inclut les informations relatives aux dernières utilisations de clé d'accès pour tous les utilisateurs IAM de votre compte.
4. (facultatif) Sélectionnez l'utilisateur IAM pour afficher les détails de l'utilisateur. La section Récapitulatif inclut les ID de clé d'accès, leur statut et la date de leur dernière utilisation.

AWS CLI

Exécutez la commande suivante :

- `aws iam get-access-key-last-used`

API

Appelez l’opération suivante :

- `GetAccessKeyLastUsed`

Mise à jour des clés d’accès

Comme bonne pratique en matière de sécurité, nous vous recommandons de mettre à jour les clés d'accès de l'utilisateur IAM en cas de besoin, par exemple lorsqu'un employé quitte votre entreprise. Les utilisateurs IAM peuvent mettre à jour leurs propres clés d'accès s'ils ont obtenu les autorisations nécessaires.

Pour plus d'informations concernant l'octroi aux utilisateurs IAM d'autorisations de mise à jour de leurs propres clés d'accès, veuillez consulter AWS : autorise les utilisateurs IAM à gérer leurs propres mot de passe, clés d’accès et clés publiques SSH sur la page Informations d’identification de sécurité. Vous pouvez également appliquer une politique de mots de passe à votre compte pour exiger que tous vos utilisateurs IAM mettent périodiquement à jour leurs mots de passe et la fréquence à laquelle ils doivent le faire. Pour de plus amples informations, veuillez consulter Définition d’une politique de mot de passe du compte pour les utilisateurs IAM.

> **Note**
>
> Si vous perdez votre clé d'accès secrète, vous devez supprimer la clé d'accès et en créer une nouvelle. La clé d’accès secrète ne peut être récupérée qu’au moment de sa création. Utilisez cette procédure pour désactiver puis remplacer les clés d’accès perdues par de nouvelles informations d’identification.

Rubriques

- Mise à jour des clés d'accès pour un utilisateur IAM (console)
- Mise à jour des clés d'accès (AWS CLI)
- Mise à jour des clés d'accès (AWS API)

Mise à jour des clés d'accès pour un utilisateur IAM (console)

Vous pouvez mettre à jour les clés d'accès à partir de l' Console de gestion AWS.

Pour mettre à jour les clés d'accès pour un utilisateur IAM sans interrompre vos applications (console)

1. Pendant que la première clé d'accès est encore active, créez une seconde clé d'accès.

   a. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.

   b. Dans le panneau de navigation, choisissez utilisateurs.

   c. Choisissez le nom utilisateur concerné, puis sélectionnez l'onglet Informations d'identification de sécurité.

   d. Dans la section Clés d'accès, choisissez Créer une clé d'accès. Sur la page des bonnes pratiques et alternatives en matière de clés d'accès, choisissez Other (Autre), puis Next (Suivant).

   e. (Facultatif) Définissez une valeur de balise de description pour la clé d'accès afin d'ajouter une paire clé-valeur de balise à cet utilisateur IAM. Cela peut vous aider à identifier et à mettre à jour les clés d'accès par la suite. La clé de balise est définie sur l'identifiant de la clé d'accès. La valeur de balise est définie selon la description de la clé d'accès que vous spécifiez. Lorsque vous avez terminé, sélectionnez Create access key (Créer la clé d'accès).

   f. Sur la page Retrieve access keys (Récupérer les clés d'accès), choisissez Show (Afficher) (pour révéler la valeur de la clé d'accès secrète de votre utilisateur) ou Download .csv file (Télécharger le fichier .csv). C'est le seul moment où vous pouvez enregistrer votre clé d'accès secrète. Après avoir enregistré votre clé d'accès secrète dans un emplacement sécurisé, sélectionnez Done (Terminé).

      Lorsque vous créez une clé d'accès pour votre utilisateur, cette paire de clés est activée par défaut et votre utilisateur peut immédiatement l'utiliser. À ce stade, l'utilisateur dispose de deux clés d'accès actives.

2. Mettez à jour toutes les applications et tous les outils pour utiliser la nouvelle clé d'accès.

3. Déterminez si la première clé d'accès est toujours utilisée en consultant les informations Dernière utilisation de la clé d'accès la plus ancienne. Une des approches possibles consiste à attendre plusieurs jours, puis à vérifier si l'ancienne clé d'accès a été utilisée avant de poursuivre.

4. Même si la valeur des informations Dernière utilisation indique que l'ancienne clé n'a jamais été utilisée, nous vous recommandons de ne pas supprimer immédiatement la première clé d'accès.

À la place, sélectionnez Actions, puis Deactivate (Désactiver) pour désactiver la première clé d'accès.

5. Utilisez uniquement la nouvelle clé d'accès pour confirmer que vos applications fonctionnent. Toutes les applications et tous les outils qui utilisent encore la clé d'accès d'origine cesseront de fonctionner à ce stade car ils n'ont plus accès aux AWS ressources. Si vous rencontrez l'une de ces applications ou l'un de ces outils, vous pouvez réactiver la première clé d'accès. Revenez ensuite à Step 3 et mettez à jour cette application afin d'utiliser la nouvelle clé.
6. Après avoir attendu un certain temps pour vous assurer que toutes les applications et tous les outils ont été mis à jour, vous pouvez supprimer la première clé d'accès :

   a. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.

   b. Dans le panneau de navigation, choisissez utilisateurs.

   c. Choisissez le nom utilisateur concerné, puis sélectionnez l'onglet Informations d'identification de sécurité.

   d. Dans la section Access keys (Clés d'accès) correspondant à la clé d'accès que vous souhaitez supprimer, choisissez Actions, puis Delete (Supprimer). Suivez les instructions de la boîte de dialogue pour tout d'abord désactiver, puis confirmer la suppression.

Pour déterminer quelles clés d'accès doivent être mises à jour ou supprimées (console)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Au besoin, ajoutez la colonne Access key age (Âge de la clé d'accès) à la table des utilisateurs en procédant comme suit :

   a. Au-dessus de la table à l'extrême droite, choisissez l'icône des paramètres ( ).

   b. Dans Manage columns (Gérer les colonnes), sélectionnez Access key age (Âge de la clé d'accès).

   c. Choisissez Fermer pour revenir à la liste des utilisateurs.
4. La colonne Access key age (Âge de la clé d'accès) affiche le nombre de jours écoulés depuis la création de la clé d'accès active la plus ancienne. Vous pouvez utiliser ces informations pour

trouver les utilisateurs dont les clés d'accès doivent être mises à jour ou supprimées. La colonne affiche Aucun pour les utilisateurs sans clé d'accès.

Mise à jour des clés d'accès (AWS CLI)

Vous pouvez mettre à jour les clés d'accès à partir de l' AWS Command Line Interface.

Pour mettre à jour les clés d'accès sans interrompre vos applications (AWS CLI)

1. Pendant que la première clé d'accès est encore active, créez une seconde clé d'accès qui est active par défaut. Exécutez la commande suivante :

   - `aws iam create-access-key`

   À ce stade, l'utilisateur dispose de deux clés d'accès actives.
2. Mettez à jour toutes les applications et tous les outils pour utiliser la nouvelle clé d'accès.
3. Déterminez si la première clé d'accès est toujours utilisée à l'aide de cette commande :

   - `aws iam get-access-key-last-used`

   Une des approches possibles consiste à attendre plusieurs jours, puis à vérifier si l'ancienne clé d'accès a été utilisée avant de poursuivre.
4. Même si l'étape Step 3 n'indique aucune utilisation de l'ancienne clé, nous vous recommandons de ne pas supprimer immédiatement la première clé d'accès. Définissez plutôt l'état de la première clé d'accès sur `Inactive` à l'aide de cette commande :

   - `aws iam update-access-key`
5. Utilisez uniquement la nouvelle clé d'accès pour confirmer que vos applications fonctionnent. Toutes les applications et tous les outils qui utilisent encore la clé d'accès d'origine cesseront de fonctionner à ce stade car ils n'ont plus accès aux AWS ressources. Si vous rencontrez l'une de ces applications ou l'un de ces outils, vous pouvez définir à nouveau son état sur `Active` pour réactiver la première clé d'accès. Revenez ensuite à l'étape Step 2 et mettez à jour cette application afin d'utiliser la nouvelle clé.
6. Après avoir attendu un certain temps pour vous assurer que toutes les applications et tous les outils ont été mis à jour, vous pouvez supprimer la première clé d'accès à l'aide de cette commande :

   - `aws iam delete-access-key`

Mise à jour des clés d'accès (AWS API)

Vous pouvez mettre à jour les clés d'accès à l'aide de AWS l'API.

Pour mettre à jour les clés d'accès sans interrompre vos applications (AWS API)

1. Pendant que la première clé d'accès est encore active, créez une seconde clé d'accès qui est active par défaut. Appelez l'opération suivante :

   - `CreateAccessKey`

   À ce stade, l'utilisateur dispose de deux clés d'accès actives.
2. Mettez à jour toutes les applications et tous les outils pour utiliser la nouvelle clé d'accès.
3. Déterminez si la première clé d'accès est toujours utilisée en appelant cette opération :

   - `GetAccessKeyLastUsed`

   Une des approches possibles consiste à attendre plusieurs jours, puis à vérifier si l'ancienne clé d'accès a été utilisée avant de poursuivre.
4. Même si l'étape Step 3 n'indique aucune utilisation de l'ancienne clé, nous vous recommandons de ne pas supprimer immédiatement la première clé d'accès. Modifiez plutôt l'état de la première clé d'accès `Inactive` en appelant cette opération :

   - `UpdateAccessKey`
5. Utilisez uniquement la nouvelle clé d'accès pour confirmer que vos applications fonctionnent. Toutes les applications et tous les outils qui utilisent encore la clé d'accès d'origine cesseront de fonctionner à ce stade car ils n'ont plus accès aux AWS ressources. Si vous rencontrez l'une de ces applications ou l'un de ces outils, vous pouvez définir à nouveau son état sur `Active` pour réactiver la première clé d'accès. Revenez ensuite à l'étape Step 2 et mettez à jour cette application afin d'utiliser la nouvelle clé.
6. Après avoir attendu un certain temps pour vous assurer que toutes les applications et tous les outils ont été mis à jour, vous pouvez supprimer la première clé d'accès en effectuant cette opération :

   - `DeleteAccessKey`

Sécurisation des clés d'accès

Toute personne disposant de vos clés d'accès dispose du même niveau d'accès à vos AWS ressources que vous. Par conséquent, AWS met tout en œuvre pour protéger vos clés d'accès et, conformément à notre modèle de responsabilité partagée, vous devriez également le faire.

Développez les sections suivantes pour obtenir des conseils qui vous aideront à protéger vos clés d'accès.

> **Note**
>
> Votre organisation peut avoir des exigences et des politiques de sécurité différentes de celles décrites dans cette rubrique. Les suggestions fournies ici doivent être considérées comme des directives générales.

Supprimer (ou ne pas générer) Utilisateur racine d'un compte AWS clés d'accès

Le meilleur moyen de protéger votre compte est de n'avoir aucune clé d'accès pour votre Utilisateur racine d'un compte AWS. À moins d'avoir impérativement besoin des clés d'accès de l'utilisateur root (ce qui est rare), il est préférable de ne pas les générer. Créez plutôt un utilisateur administratif pour les tâches administratives quotidiennes. AWS IAM Identity Center Pour plus d'informations sur la création d'un utilisateur administratif dans IAM Identity Center, consultez la section https://docs.aws.amazon.com/singlesignon/latest/userguide/getting-started.html Démarrage dans le Guide de l'utilisateur d'IAM Identity Center.

Si vous disposez déjà de clés d'accès d'utilisateur root pour votre compte, nous vous recommandons ce qui suit : recherchez dans vos applications les endroits où vous utilisez actuellement des clés d'accès (le cas échéant) et remplacez les clés d'accès de l'utilisateur root par celles de l'utilisateur IAM. Ensuite, désactivez et supprimez les clés d'accès de l'utilisateur root. Pour plus d'informations sur les modalités de mise à jour des clés d'accès, veuillez consulter Mise à jour des clés d'accès.

Utiliser les informations d'identification de sécurité temporaires (rôles IAM) au lieu des clés d'accès à long terme

Dans de nombreux cas, vous n'avez pas besoin d'une clé d'accès à long terme qui n'expire jamais (comme dans le cas d'un utilisateur IAM). Au lieu de cela, vous pouvez créer des rôles IAM et générer des informations d'identification de sécurité temporaires. Les informations d'identification

de sécurité temporaires consistent en un ID de clé d'accès et une clé d'accès secrète, mais elles comprennent également un jeton de sécurité qui indique la date d'expiration des informations d'identification.

Long-term les clés d'accès, telles que celles associées aux utilisateurs IAM et à l'utilisateur root, restent valides jusqu'à ce que vous les révoquiez manuellement. Cependant, les informations d'identification de sécurité temporaires obtenues via les rôles IAM et d'autres fonctionnalités AWS Security Token Service expirent après une courte période. Utilisez les informations d'identification de sécurité temporaires pour vous aider à réduire les risques en cas de compromission accidentelle des informations d'identification.

Utilisez un rôle IAM et les informations d'identification de sécurité temporaires dans les cas suivants :

- Vous avez une application ou AWS CLI des scripts qui s'exécutent sur une instance Amazon EC2. N'utilisez pas les clés d'accès directement dans votre application. Ne transmettez pas les clés d'accès à l'application, ne les intégrez pas à l'application et ne laissez pas l'application les lire depuis n'importe quelle source. Définissez plutôt un rôle IAM qui dispose des autorisations adéquates pour votre application et lancez l'instance Amazon Elastic Compute Cloud (Amazon EC2) avec des rôles pour EC2. Cela permet d'associer un rôle IAM à l'instance Amazon EC2. Cette pratique permet également à l'application d'obtenir des informations d'identification de sécurité temporaires qu'elle peut à son tour utiliser pour effectuer des appels à AWS. Les AWS SDK et AWS Command Line Interface (AWS CLI) peuvent obtenir automatiquement des informations d'identification temporaires à partir du rôle.
- Vous devez accorder l'accès entre comptes. Utilisez un rôle IAM pour établir une relation d'approbation entre les comptes, puis accordez aux utilisateurs d'un compte des autorisations limitées pour accéder au compte approuvé. Pour de plus amples informations, veuillez consulter Tutoriel IAM : accès délégué via AWS comptes utilisant des rôles IAM.
- Vous disposez d'une application mobile. N'intégrez pas les clés d'accès à l'application, même dans un espace de stockage chiffré. Au lieu de cela, utilisez Amazon Cognito pour gérer les identités de l'utilisateur dans votre application. Ce service vous permet d'authentifier les utilisateurs à l'aide de Login with Amazon, Facebook, Google ou tout autre fournisseur d'identité compatible avec OpenID Connect (OIDC). Vous pouvez ensuite utiliser le fournisseur d'informations d'identification d'Amazon Cognito pour gérer les informations d'identification que votre application utilise pour effectuer des requêtes à AWS.
- Vous souhaitez vous fédérer AWS et votre organisation prend en charge le protocole SAML 2.0. Si vous travaillez pour une organisation disposant d'un fournisseur d'identité prenant en charge SAML 2.0, configurez le fournisseur pour qu'il utilise SAML. Vous pouvez utiliser SAML pour échanger

des informations d'authentification AWS et récupérer un ensemble d'informations d'identification de sécurité temporaires. Pour de plus amples informations, veuillez consulter Fédération SAML 2.0.

- Vous souhaitez vous fédérer au sein d'une banque d'identités sur site AWS et votre organisation dispose d'une banque d'identités sur site. Si les utilisateurs peuvent s'authentifier au sein de votre organisation, vous pouvez créer une application qui peut leur délivrer des informations d'identification de sécurité temporaires pour accéder aux AWS ressources. Pour de plus amples informations, veuillez consulter Activez l'accès personnalisé du courtier d'identité au AWS console.
- Utilisez les conditions des politiques IAM pour n'autoriser l'accès qu'à partir des réseaux attendus. Vous pouvez limiter où et comment vos clés d'accès sont utilisées en mettant en œuvre des politiques IAM avec des conditions qui spécifient et n'autorisent que les réseaux attendus, tels que vos adresses IP publiques ou vos clouds privés virtuels (VPC). De cette façon, vous savez que les clés d'accès ne peuvent être utilisées qu'à partir de réseaux attendus et acceptables.

> **Note**
>
> Utilisez-vous une instance Amazon EC2 avec une application qui nécessite un accès programmatique aux ressources ? AWS Dans ce cas, utilisez les rôles IAM pour EC2.

Gérer correctement les clés d'accès d'un utilisateur IAM

Si vous devez créer des clés d'accès pour l'accès par programmation AWS, créez-les pour les utilisateurs IAM, en leur accordant uniquement les autorisations dont ils ont besoin.

Respectez les précautions suivantes pour protéger les clés d'accès des utilisateurs IAM :

- N'intégrez pas de clés d'accès directement dans le code. Les kits AWS SDK et les outils de ligne de commande AWS vous permettent de placer les clés d'accès à des emplacements connus, afin que vous n'ayez pas à les conserver dans le code.

  Placez les clés d'accès à l'un des emplacements suivants :

  - Le fichier AWS d'informations d'identification. Les AWS kits SDK utilisent AWS CLI automatiquement les informations d'identification que vous stockez dans le fichier AWS d'informations d'identification.

    Pour plus d'informations sur l'utilisation du fichier AWS d'informations d'identification, consultez la documentation de votre SDK. Les exemples incluent Définir les AWS informations d'identification et la région dans le Guide du AWS SDK pour Java développeur et les fichiers

de configuration et d'informations d'identification dans le Guide de AWS Command Line Interface l'utilisateur.

Pour stocker les informations d'identification AWS SDK pour .NET des et des AWS Tools for Windows PowerShell, nous vous recommandons d'utiliser le SDK Store. Pour plus d'informations, veuillez consulter la rubrique Using the SDK Store dans le Guide du développeur AWS SDK pour .NET .

- Variables d'environnement. Sur un système à locataires multiples, choisissez des variables d'environnement utilisateur, et non pas des variables d'environnement système.

  Pour plus d'informations sur l'utilisation des variables d'environnement pour stocker les informations d'identification, veuillez consulter la rubrique Environment Variables dans le Guide de l'utilisateur AWS Command Line Interface .

- Utilisez des clés d'accès différentes pour chaque application. Procédez ainsi afin de pouvoir isoler les autorisations et révoquer les clés d'accès pour des applications individuelles si elles sont compromises. Le fait d'avoir des clés d'accès distinctes pour différentes applications génère également des entrées distinctes dans les fichiers journaux AWS CloudTrail. Cette configuration vous permet d'identifier plus facilement quelle application a effectué des actions spécifiques.
- Mettez à jour les clés d'accès en cas de besoin. Si la clé d'accès risque d'être compromise, mettez-la à jour et supprimez la clé d'accès précédente. Pour plus d'informations, consultez Mise à jour des clés d'accès.
- Supprimez les clés d'accès inutilisées. Si un utilisateur quitte votre organisation, supprimez l'utilisateur IAM correspondant afin qu'il ne puisse plus accéder à vos ressources. Pour savoir quand une clé d'accès a été utilisée pour la dernière fois, utilisez l'`GetAccessKeyLastUsed`API (AWS CLI commande : `aws iam get-access-key-last-used`).
- Utilisez des informations d'identification temporaires et configurez l'authentification multifactorielle pour vos opérations d'API les plus sensibles. Avec les politiques IAM, vous pouvez spécifier quelles opérations d'API un utilisateur est autorisé à appeler. Dans certains cas, vous souhaiterez peut-être bénéficier d'une sécurité supplémentaire en obligeant les utilisateurs à s'authentifier auprès de l'authentification AWS MFA avant de les autoriser à effectuer des actions particulièrement sensibles. Par exemple, vous disposez peut-être d'une politique qui autorise l'utilisateur à exécuter les actions Amazon EC2 `RunInstances`, `DescribeInstances` et `StopInstances`. Mais vous souhaiterez peut-être restreindre une action destructrice, par exemple, `TerminateInstances` et vous assurer que les utilisateurs ne peuvent effectuer cette action que s'ils s'authentifient auprès d'un appareil AWS MFA. Pour de plus amples informations, veuillez consulter Accès sécurisé aux API avec MFA.

Accédez à l'application mobile en utilisant AWS clés d'accès

Vous pouvez accéder à un ensemble limité de AWS services et de fonctionnalités à l'aide de l'application AWS mobile. L'application mobile vous aide à prendre en charge la réponse aux incidents pendant vos déplacements. Pour de plus amples informations et pour télécharger l'application, veuillez consulter Console AWS pour les applications mobiles.

Vous pouvez vous connecter à l'application mobile à l'aide du mot de passe de votre console ou de vos clés d'accès. En guise de bonne pratique, n'utilisez pas les clés d'accès de l'utilisateur root. Nous vous recommandons vivement, en plus d'utiliser un mot de passe ou un verrou biométrique sur votre appareil mobile, de créer un utilisateur IAM spécifiquement chargé de gérer les AWS ressources à l'aide de l'application mobile. Si vous perdez votre appareil mobile, vous pouvez supprimer l'accès de l'utilisateur IAM.

Pour vous connecter à l'aide des clés d'accès (application mobile)

1. Ouvrez l'application sur votre appareil mobile.
2. Si c'est la première fois que vous ajoutez une identité à l'appareil, choisissez Ajouter une identité, puis cliquez sur Clés d'accès.

   Si vous vous êtes déjà connecté à l'aide d'une autre identité, choisissez l'icône de menu et choisissez Changer d'identité. Choisissez ensuite Se connecter en tant qu'identité différente, puis Clés d'accès.
3. Sur la page Clés d'accès, saisissez vos informations :

   - ID de clé d'accès : saisissez votre ID de clé d'accès.
   - Clé d'accès secrète : saisissez votre clé d'accès secrète.
   - Nom de l'identité : saisissez le nom de l'identité qui apparaîtra dans l'application mobile. Elle ne doit pas nécessairement correspondre à votre nom d'utilisateur IAM.
   - Code PIN d'identité : créez un numéro d'identification personnel (PIN) que vous utiliserez pour les prochaines connexions.

   > **Note**
   >
   > Si vous activez la biométrie pour l'application AWS mobile, vous serez invité à utiliser votre empreinte digitale ou votre reconnaissance faciale à des fins de vérification au

lieu du code PIN. Si la biométrie échoue, vous pouvez être invité à entrer le code PIN à la place.

4. Choisissez Vérifier et ajouter des clés.

   Vous pouvez désormais accéder à un ensemble sélectionné de vos ressources à l'aide de l'application mobile.

Informations connexes

Les rubriques suivantes fournissent des conseils pour configurer les AWS kits de développement logiciel et AWS CLI utiliser les clés d'accès :

- Définissez AWS les informations d'identification et la région dans le Guide du AWS SDK pour Java développeur
- Using the SDK Store dans le Guide du développeur AWS SDK pour .NET
- Providing Credentials to the SDK dans le Guide du développeur AWS SDK pour PHP
- Configuration dans la documentation Boto 3 (AWS SDK pour Python)
- Using AWS Credentials dans le Guide de l'utilisateur AWS Tools for Windows PowerShell
- Configuration and credential files dans le Guide de l'utilisateur AWS Command Line Interface
- Granting access using an IAM role dans le Guide du développeur AWS SDK pour .NET
- Configure IAM roles for Amazon EC2 dans le AWS SDK for Java 2.x

Utilisation des clés d'accès et des informations d'identification de clé secrète pour l'accès à la console

Il est possible d'utiliser des informations d'identification de clé d'accès et de clé secrète pour un accès à l' Console de gestion AWS direct, et pas seulement l' AWS CLI. Cela peut être réalisé à l'aide de l'appel AWS STS `GetFederationToken` d'API. En créant une URL de console à l'aide des informations d'identification temporaires et du jeton fournis par `GetFederationToken`, les principaux IAM peuvent accéder à la console. Pour de plus amples informations, veuillez consulter Activez l'accès personnalisé du courtier d'identité au AWS console.

Il convient de préciser que lorsque vous vous connectez directement à la console à l'aide des informations d'identification IAM ou utilisateur racine avec la MFA activée, cette dernière sera requise. Toutefois, si la méthode décrite ci-dessus (en utilisant des informations d'identification temporaires avec `GetFederationToken`) est utilisée, la MFA ne sera PAS requise.

Audit des clés d'accès

Vous pouvez consulter les clés AWS d'accès contenues dans votre code pour déterminer si elles proviennent d'un compte que vous possédez. Vous pouvez transmettre un identifiant de clé d'accès à l'aide de la `aws sts get-access-key-info` AWS CLI commande ou de l'opération `GetAccessKeyInfo` AWS API.

Les opérations AWS d'API AWS CLI et renvoient l'ID Compte AWS auquel appartient la clé d'accès. Les ID de clé d'accès commençant par `AKIA` sont les informations d'identification à long terme d'un utilisateur IAM ou d'un Utilisateur racine d'un compte AWS. Les ID de clé d'accès commençant par `ASIA` sont des informations d'identification temporaires créées à l'aide d' AWS STS opérations. Si le compte de la réponse vous appartient, vous pouvez vous connecter en tant qu'utilisateur racine et vérifier vos clés d'accès d'utilisateur racine. Ensuite, vous pouvez extraire un rapport d'informations d'identification pour savoir quel utilisateur IAM possède les clés. Pour savoir qui a demandé les informations d'identification temporaires pour une clé d'`ASIA`accès, consultez les AWS STS événements dans vos CloudTrail journaux.

Pour des raisons de sécurité, vous pouvez consulter AWS CloudTrail les journaux pour savoir qui a effectué une action AWS. Vous pouvez utiliser la clé de condition `sts:SourceIdentity` dans la politique d'approbation de rôle pour exiger des utilisateurs qu'ils spécifient une identité lorsqu'ils endossent un rôle. Par exemple, vous pouvez exiger que les utilisateurs IAM spécifient leur propre nom d'utilisateur comme identité de source. Cela peut vous aider à déterminer quel utilisateur a effectué une action spécifique dans AWS. Pour de plus amples informations, veuillez consulter `sts:SourceIdentity`.

Cette opération n'indique pas l'état de la clé d'accès. La clé peut être active, inactive ou supprimée. Les clés actives peuvent ne pas avoir les autorisations nécessaires pour effectuer une opération. La fourniture d'une clé d'accès supprimée peut renvoyer une erreur indiquant que la clé n'existe pas.

AWS Multi-factor authentification dans IAM

Pour renforcer la sécurité, nous vous recommandons de configurer l'authentification multifacteur (MFA) afin de protéger vos AWS ressources. Vous pouvez activer l' Utilisateur racine d'un compte AWS authentification multifacteur pour tous Comptes AWS, y compris les comptes autonomes, les comptes de gestion et les comptes de membres, ainsi que pour vos utilisateurs IAM. Nous vous recommandons d'utiliser une authentification multifacteur résistante au phishing, telle que des clés d'accès et des clés de sécurité, dans la mesure du possible. Ces FIDO-based authentificateurs

utilisent la cryptographie à clé publique et résistent au phishing, aux attaques de type man-in-the-middle et aux attaques par replay, offrant ainsi un niveau de sécurité supérieur à celui des options. TOTP-based

La MFA est appliquée à tous les types de comptes pour leur utilisateur racine. Pour de plus amples informations, veuillez consulter Sécurisez votre AWS Organizations informations d'identification de l'utilisateur root du compte.

Lorsque vous activez l'authentification MFA pour l'utilisateur root, cela affecte uniquement les informations d'identification de l'utilisateur root. Les utilisateurs IAM du compte sont des identités distinctes avec leurs propres informations d'identification, et chaque identité dispose de sa propre configuration MFA. Pour de plus amples informations sur l'utilisation de l'authentification multifactorielle (MFA) pour protéger l'utilisateur racine, veuillez consulter Multi-factor authentification pour Utilisateur racine d'un compte AWS.

Vos utilisateurs Utilisateur racine d'un compte AWS et ceux d'IAM peuvent enregistrer jusqu'à huit appareils MFA de tout type. L'enregistrement de plusieurs dispositifs MFA peut apporter de la flexibilité et vous aider à réduire le risque d'interruption d'accès en cas de perte ou de panne d'un dispositif. Vous n'avez besoin que d'un seul dispositif MFA pour vous connecter à l' Console de gestion AWS ou créer une session via l' AWS CLI.

> **Note**
>
> Nous vous recommandons de demander à vos utilisateurs humains d'utiliser des informations d'identification temporaires lors de l'accès AWS. Avez-vous envisagé de l'utiliser AWS IAM Identity Center ? Vous pouvez utiliser IAM Identity Center pour gérer de manière centralisée l'accès à plusieurs comptes Comptes AWS et fournir aux MFA-protected utilisateurs un accès en authentification unique à tous les comptes qui leur sont attribués depuis un seul endroit. Avec IAM Identity Center, vous pouvez créer et gérer les identités des utilisateurs dans IAM Identity Center ou vous connecter facilement à votre fournisseur d'identité compatible SAML 2.0 existant. Pour plus d'informations, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Le MFA ajoute une sécurité supplémentaire qui oblige les utilisateurs à fournir une authentification unique à partir d'un mécanisme MFA AWS pris en charge en plus de leurs informations de connexion lorsqu'ils accèdent à des AWS sites Web ou à des services.

Types MFA

AWS prend en charge les types de MFA suivants :

Table des matières

- Clés d’accès et clés de sécurité
- Applications d’authentification virtuelle
- Jetons TOTP matériels

Clés d’accès et clés de sécurité

Gestion des identités et des accès AWS prend en charge les clés d'accès et les clés de sécurité pour le MFA. Sur la base des normes FIDO, les clés d'accès utilisent la cryptographie à clé publique pour fournir une authentification robuste et résistante au hameçonnage, qui est plus sûre que les mots de passe. AWS prend en charge deux types de clés d'accès : les clés d'accès liées à l'appareil (clés de sécurité) et les clés d'accès synchronisées.

- Clés de sécurité : il s'agit de périphériques physiques YubiKey, comme un, utilisés comme second facteur d'authentification. Une clé de sécurité unique peut prendre en charge plusieurs comptes utilisateur racine et utilisateurs IAM.
- Clés d’accès synchronisées : elles utilisent des gestionnaires d’informations d’identification provenant de fournisseurs tels que Google, Apple, Microsoft et de services tiers tels que 1Password, Dashlane et Bitwarden comme deuxième facteur.

Vous pouvez utiliser des authentificateurs biométriques intégrés, tels que Touch ID sur Apple MacBooks, pour déverrouiller votre gestionnaire d'informations d'identification et vous y connecter. AWS Les clés d’accès sont créées avec le fournisseur de votre choix à l’aide de votre empreinte digitale, de votre visage ou du code PIN de votre appareil. Vous pouvez utiliser une clé d’accès d’authentification entre appareils (CDA) provenant d’un appareil, comme un appareil mobile ou une clé de sécurité matérielle, pour vous connecter sur un autre appareil, tel qu’un ordinateur portable. Pour plus d’informations, consultez Authentification entre appareils (CDA).

Vous pouvez synchroniser les clés d'accès sur tous vos appareils pour faciliter les connexions et améliorer ainsi la facilité d'utilisation et la AWS capacité de restauration. Pour plus d’informations sur l’activation des clés d’accès et des clés de sécurité, consultez Activation d’une clé d’accès ou d’une clé de sécurité pour l’utilisateur racine (console).

La FIDO Alliance tient à jour une liste de tous les produits certifiés FIDO qui sont compatibles avec les spécifications FIDO.

Applications d'authentification virtuelle

Une application d'authentification virtuelle s'exécute sur un téléphone ou un autre dispositif et émule un dispositif physique. Les applications d'authentification virtuelle mettent en œuvre l'algorithme TOTP (mot de passe unique à durée limitée) et prennent en charge plusieurs jetons sur un seul dispositif. L'utilisateur doit saisir un code valide à partir du dispositif lorsqu'il y est invité lors de la connexion. Chaque jeton attribué à un utilisateur doit être unique. Un utilisateur ne peut pas saisir un code provenant du jeton d'un autre utilisateur pour s'authentifier.

Nous vous recommandons d'utiliser une authentification multifacteur résistante au phishing, telle que des clés d'accès ou des clés de sécurité, pour une protection optimale. Si vous n'êtes pas encore en mesure d'utiliser des clés d'accès ou des clés de sécurité, nous vous recommandons d'utiliser un appareil MFA virtuel à titre de mesure provisoire en attendant l'approbation de l'achat du matériel ou en attendant l'arrivée de votre matériel. Pour obtenir la liste de quelques applications prises en charge que vous pouvez utiliser en tant que périphériques MFA virtuels, consultez Multi-Factor Authentification (MFA).

Pour plus d'informations sur la configuration d'un dispositif MFA virtuel pour un utilisateur IAM, consultez Attribuez un périphérique MFA virtuel dans le Console de gestion AWS.

> **Note**
>
> Les appareils MFA virtuels non attribués dans votre Compte AWS compte sont supprimés lorsque vous ajoutez de nouveaux appareils MFA virtuels via Console de gestion AWS ou pendant le processus de connexion. Les dispositifs MFA virtuels non attribués sont des dispositifs présents dans votre compte, mais qui ne sont pas utilisés par l'utilisateur racine du compte ou les utilisateurs IAM pour le processus de connexion. Ils sont supprimés afin que de nouveaux dispositifs MFA virtuels puissent être ajoutés à votre compte. Cela vous permet également de réutiliser les noms des appareils.
>
> - Pour afficher les appareils MFA virtuels non attribués dans votre compte, vous pouvez utiliser la commande https://awscli.amazonaws.com/v2/documentation/api/latest/reference/iam/list-virtual-mfa-devices.html AWS CLI list-virtual-mfa-devices ou un appel d'API. https://docs.aws.amazon.com/IAM/latest/APIReference/API_ListVirtualMFADevices.html
> - Pour désactiver un appareil MFA virtuel, vous pouvez utiliser la commande https://awscli.amazonaws.com/v2/documentation/api/latest/reference/iam/deactivate-

mfa-device.html AWS CLI deactivate-mfa-device ou un appel d'API. https://docs.aws.amazon.com/IAM/latest/APIReference/API_DeactivateMFADevice.html Le dispositif ne sera plus attribué.

- Pour associer un appareil MFA virtuel non attribué à votre utilisateur Compte AWS root ou à vos utilisateurs IAM, vous aurez besoin du code d'authentification généré par l'appareil ainsi que de la commande enable-mfa-device ou de l'appel AWS CLI d'API. https://docs.aws.amazon.com/IAM/latest/APIReference/API_EnableMFADevice.html

Jetons TOTP matériels

Un dispositif matériel qui génère un code numérique à six chiffres basé sur l'algorithme TOTP (mot de passe unique à durée limitée). L'utilisateur doit saisir un code valide à partir du dispositif sur une deuxième page web lors de la connexion.

Ces jetons sont utilisés exclusivement avec Comptes AWS. Vous ne pouvez utiliser que des jetons dont l'origine unique est partagée en toute sécurité AWS. Les graines de jetons sont des clés secrètes générées au moment de la production des jetons. Les jetons achetés auprès d'autres sources ne fonctionneront pas avec IAM. Pour garantir la compatibilité, vous devez acheter votre dispositif MFA matériel via l'un des liens suivants : jeton OTP ou carte d'affichage OTP.

- Chaque dispositif MFA attribué à un utilisateur doit être unique. Un utilisateur ne peut pas saisir un code à partir du périphérique d'un autre utilisateur pour s'authentifier. Pour plus d'informations sur les périphériques MFA matériels pris en charge, voir Multi-Factor Authentification (MFA).
- Si vous souhaitez utiliser un dispositif MFA physique, nous vous recommandons d'utiliser des clés de sécurité comme alternative aux dispositifs TOTP matériels. Les clés de sécurité ne nécessitent aucune pile, résistent au hameçonnage et prennent en charge plusieurs utilisateurs sur un seul dispositif.

Vous pouvez activer une clé d'accès ou une clé de sécurité depuis la Console de gestion AWS seule API, et non depuis l' AWS API AWS CLI ou. Avant d'activer une clé de sécurité, vous devez avoir un accès physique au dispositif.

Pour plus d'informations sur la configuration d'un jeton TOTP matériel pour un utilisateur IAM, consultez Attribuez un jeton TOTP matériel dans Console de gestion AWS.

> **Note**
>
> MFA par SMS : AWS fin de la prise en charge de l'activation de l'authentification multifactorielle par SMS (MFA). Nous recommandons aux clients dont les utilisateurs IAM utilisent un dispositif MFA basé sur les SMS de passer à l'une des méthodes alternatives suivantes : clé d'accès ou clé de sécurité, dispositif MFA (logiciel) virtuel ou dispositif MFA matériel. Vous pouvez identifier les utilisateurs dans votre compte avec un dispositif MFA SMS affecté. Dans la console IAM, sélectionnez Users (Utilisateurs) dans le panneau de navigation, puis recherchez les utilisateurs avec SMS mentionné dans la colonne MFA de la table.

Recommandations MFA

Pour vous aider à sécuriser vos AWS identités, suivez ces recommandations pour l'authentification MFA.

- Nous vous recommandons d'utiliser une authentification multifacteur résistante au phishing, telle que des clés d'accès et des clés de sécurité, comme appareil MFA. Ces FIDO-based authentificateurs offrent la meilleure protection contre les attaques telles que le phishing.
- Nous vous recommandons d'activer plusieurs appareils MFA pour les utilisateurs Utilisateur racine d'un compte AWS et les utilisateurs IAM de votre. Comptes AWS Cela vous permet de renforcer la barre de sécurité Comptes AWS et de simplifier la gestion de l'accès aux utilisateurs hautement privilégiés, tels que Utilisateur racine d'un compte AWS.
- Vous pouvez enregistrer jusqu'à huit appareils MFA de n'importe quelle combinaison des types MFA actuellement pris en charge auprès de vos utilisateurs Utilisateur racine d'un compte AWS et de ceux d'IAM. Avec plusieurs appareils MFA, vous n'avez besoin que d'un seul appareil MFA pour vous connecter Console de gestion AWS ou créer une session en AWS CLI tant qu'utilisateur. Un utilisateur IAM doit s'authentifier avec un dispositif MFA existant pour activer ou désactiver un dispositif MFA supplémentaire.
- En cas de perte, de vol ou d'inaccessibilité d'un appareil MFA, vous pouvez utiliser l'un des appareils MFA restants pour y accéder Compte AWS sans effectuer la procédure de Compte AWS restauration. En cas de perte ou de vol d'un dispositif MFA, celui-ci doit être dissocié du principal IAM auquel il est associé.

- L'utilisation de plusieurs MFA permet à vos employés situés sur des sites géographiquement dispersés ou travaillant à distance d'utiliser le MFA matériel pour accéder AWS sans avoir à coordonner l'échange physique d'un seul périphérique matériel entre les employés.
- L'utilisation de dispositifs MFA supplémentaires pour les principaux IAM vous permet d'utiliser un ou plusieurs MFA pour un usage quotidien, tout en conservant les dispositifs MFA physiques dans un emplacement physique sécurisé, tel qu'une chambre forte ou un coffre-fort pour la sauvegarde et la redondance.

Remarques

- Vous ne pouvez pas transmettre les informations MFA d'une clé de sécurité ou d'un mot de passe aux opérations d' AWS STS API pour demander des informations d'identification temporaires. Vous pouvez obtenir des informations d'identification à utiliser avec les AWS SDK AWS CLI et lorsque vous utilisez une clé de sécurité ou une clé d'accès en exécutant la `aws login` commande.
- Vous ne pouvez pas utiliser de AWS CLI commandes ou AWS d'opérations d'API pour activer les clés de sécurité FIDO.
- Vous ne pouvez pas utiliser le même nom pour plusieurs utilisateurs racine ou dispositif MFA IAM.

Ressources supplémentaires

Les ressources suivantes peuvent vous aider à en savoir plus au sujet de la MFA.

- Pour plus d'informations sur l'utilisation de l'authentification multifacteur pour accéder AWS, consultezConnexion compatible avec la MFA.
- Vous pouvez utiliser IAM Identity Center pour activer un accès MFA sécurisé à votre portail AWS d'accès, aux applications intégrées IAM Identity Center et au. AWS CLI Pour plus d'informations, consultez Activation de la MFA dans IAM Identity Center.
- Pour vérifier si l'authentification multifacteur est déjà activée pour vos utilisateurs, consultezVérification du statut MFA.

Attribuez un mot de passe ou une clé de sécurité dans le Console de gestion AWS

Les clés d'accès sont un type de dispositif d'authentification multifacteur (MFA) que vous pouvez utiliser pour protéger vos ressources. AWS AWS prend en charge les clés d'accès synchronisées et les clés d'accès liées à l'appareil, également appelées clés de sécurité.

Les clés d'accès synchronisées permettent aux utilisateurs d'IAM d'accéder à leurs informations d'identification de connexion FIDO sur bon nombre de leurs appareils, même les plus récents, sans avoir à réinscrire chaque appareil sur chaque compte. Les clés d'accès synchronisées incluent des gestionnaires d'identifiants internes tels que Google, Apple et Microsoft et des gestionnaires d'identifiants tiers tels que 1Password, Dashlane et Bitwarden comme deuxième facteur. Vous pouvez également utiliser la biométrie intégrée à l'appareil (par exemple, TouchID, FaceID) pour déverrouiller le gestionnaire d'informations d'identification de votre choix afin qu'il utilise des clés d'accès.

D'autre part, les clés d'accès liées à l'appareil sont liées à une clé de sécurité FIDO que vous branchez sur un port USB de votre ordinateur, puis que vous touchez lorsque vous y êtes invité pour terminer le processus de connexion en toute sécurité. Si vous utilisez déjà une clé de sécurité FIDO avec d'autres services et que sa configuration est AWS prise en charge (par exemple, la série YubiKey 5 de Yubico), vous pouvez également l'utiliser avec. AWS Sinon, vous devez acheter une clé de sécurité FIDO si vous souhaitez l'utiliser WebAuthn pour le MFA. AWS De plus, les clés de sécurité FIDO peuvent prendre en charge plusieurs utilisateurs IAM ou utilisateurs racine sur le même appareil, ce qui améliore leur utilité pour la sécurité des comptes. Pour les spécifications et les informations d'achat pour les deux types d'appareils, voir Multi-Factor Authentification.

Vous pouvez enregistrer jusqu'à huit appareils MFA de n'importe quelle combinaison des types MFA actuellement pris en charge auprès de vos utilisateurs Utilisateur racine d'un compte AWS et de ceux d'IAM. Avec plusieurs appareils MFA, vous n'avez besoin que d'un seul appareil MFA pour vous connecter Console de gestion AWS ou créer une session en AWS CLI tant qu'utilisateur. Nous vous recommandons d'enregistrer plusieurs appareils MFA. Par exemple, vous pouvez enregistrer un authentificateur intégré ainsi qu'une clé de sécurité que vous conservez dans un endroit physiquement sûr. Si vous ne pouvez pas à utiliser votre authentificateur intégré, vous pouvez utiliser votre clé de sécurité enregistrée. Pour les applications d'authentification, nous recommandons également d'activer la fonctionnalité de sauvegarde ou de synchronisation dans le cloud dans ces applications afin d'éviter de perdre l'accès à votre compte si vous perdez ou cassez votre appareil avec les applications d'authentification.

> **Note**
>
> Nous vous recommandons de demander à vos utilisateurs humains d'utiliser des informations d'identification temporaires lors de l'accès à AWS. Vos utilisateurs peuvent se fédérer AWS auprès d'un fournisseur d'identité où ils s'authentifient à l'aide de leurs informations d'identification d'entreprise et de leurs configurations MFA. Pour gérer l'accès AWS et les applications métier, nous vous recommandons d'utiliser IAM Identity Center. Pour plus d'informations, consultez le Guide de l'utilisateur IAM Identity Center.

Rubriques

- Autorisations nécessaires
- Activation d'une clé d'accès ou d'une clé de sécurité pour votre propre utilisateur IAM (console)
- Activation d'une clé d'accès ou d'une clé de sécurité pour un autre utilisateur IAM (console)
- Remplacement d'une clé d'accès ou d'une clé de sécurité
- Configurations prises en charge pour l'utilisation des clés d'accès et des clés de sécurité

Autorisations nécessaires

Pour gérer une clé d'accès FIDO pour votre propre utilisateur IAM tout en protégeant les MFA-related actions sensibles, vous devez disposer des autorisations prévues par la politique suivante :

> **Note**
>
> Les valeurs ARN sont des valeurs statiques et n'indiquent pas le protocole qui a été utilisé pour enregistrer l'authentificateur. Nous avons rendu U2F obsolète, donc toutes les nouvelles implémentations utilisent. WebAuthn

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Activation d'une clé d'accès ou d'une clé de sécurité pour votre propre utilisateur IAM (console)

Vous pouvez activer une clé d'accès ou une clé de sécurité pour votre propre utilisateur IAM à partir de la Console de gestion AWS seule API, et non de l'API AWS CLI ou AWS . Avant d'activer une clé de sécurité, vous devez avoir un accès physique au dispositif.

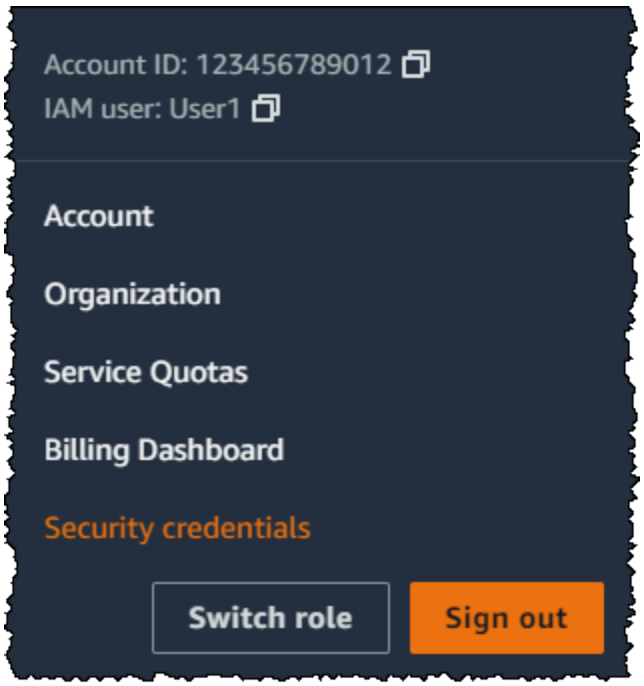
Pour activer une clé d'accès ou une clé de sécurité pour votre propre utilisateur IAM (console)

1. Utilisez votre identifiant de AWS compte ou votre alias de compte, votre nom d'utilisateur IAM et votre mot de passe pour vous connecter à la console IAM.

> **Note**
>
> Pour votre commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser votre nom d'utilisateur IAM et les informations de votre compte. Si vous vous êtes déjà connecté en tant qu'utilisateur différent, sélectionnez Sign in to a different account (Se connecter à un compte différent) en bas de la page pour revenir à la page de connexion principale. À partir de là, vous pouvez saisir votre identifiant de AWS compte ou votre alias de compte pour être redirigé vers la page de connexion utilisateur IAM de votre compte.

Pour obtenir votre Compte AWS identifiant, contactez votre administrateur.

2. Dans la barre de navigation, en haut à droite, choisissez votre nom d'utilisateur, puis Security credentials (Informations d'identification de sécurité).



3. Sur la page de l'utilisateur IAM sélectionné, choisissez l'onglet Informations d'identification de sécurité.
4. Sous Multi-factor Authentification (MFA), choisissez Attribuer un appareil MFA.
5. Sur la page Nom du dispositif MFA, saisissez un Nom du dispositif, choisissez Clé d'accès ou Clé de sécurité, puis choisissez Suivant.

6. Dans Configurer le dispositif, configurez votre clé d'accès. Créez une clé d'accès avec des données biométriques telles que votre visage ou votre empreinte digitale, avec le code PIN d'un appareil, ou en insérant la clé de sécurité FIDO dans le port USB de votre ordinateur et en la touchant.
7. Suivez les instructions de votre navigateur, puis choisissez Continuer.

Vous avez maintenant enregistré votre clé d'accès ou votre clé de sécurité à utiliser avec AWS. Pour plus d'informations sur l'utilisation de l'authentification multifacteur avec le Console de gestion AWS, consultezConnexion compatible avec la MFA.

Activation d'une clé d'accès ou d'une clé de sécurité pour un autre utilisateur IAM (console)

Vous pouvez activer une clé d'accès ou une clé de sécurité pour un autre utilisateur IAM à partir de la Console de gestion AWS seule API, et non de l'API AWS CLI ou AWS .

Pour activer une clé d'accès ou une clé de sécurité pour un autre utilisateur IAM (console)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Sous Utilisateurs, choisissez le nom de l'utilisateur pour lequel vous souhaitez activer l'authentification MFA.
4. Sur la page de l'utilisateur IAM sélectionné, choisissez l'onglet Informations d'identification de sécurité.
5. Sous Multi-factor Authentification (MFA), choisissez Attribuer un appareil MFA.
6. Sur la page Nom du dispositif MFA, saisissez un Nom du dispositif, choisissez Clé d'accès ou Clé de sécurité, puis choisissez Suivant.
7. Dans Configurer le dispositif, configurez votre clé d'accès. Créez une clé d'accès avec des données biométriques telles que votre visage ou votre empreinte digitale, avec le code PIN d'un appareil, ou en insérant la clé de sécurité FIDO dans le port USB de votre ordinateur et en la touchant.
8. Suivez les instructions de votre navigateur, puis choisissez Continuer.

Vous avez maintenant enregistré une clé d'accès ou une clé de sécurité utilisable avec AWS pour un autre utilisateur IAM. Pour plus d'informations sur l'utilisation de l'authentification multifacteur avec le Console de gestion AWS, consultezConnexion compatible avec la MFA.

Remplacement d'une clé d'accès ou d'une clé de sécurité

Jusqu'à huit appareils MFA, quelle que soit la combinaison des types MFA actuellement pris en charge, peuvent être attribués à un utilisateur à la fois avec vous Utilisateur racine d'un compte AWS et vos utilisateurs IAM. Si l'utilisateur perd un authentificateur FIDO ou a besoin de le remplacer pour une raison quelconque, vous devez tout d'abord désactiver l'ancien authentificateur FIDO. Vous pouvez alors ajouter un nouveau dispositif MFA pour l'utilisateur.

- Pour désactiver le dispositif actuellement associé à un autre utilisateur IAM, veuillez consulter la rubrique Désactivation d'un dispositif MFA.
- Pour ajouter une nouvelle clé de sécurité FIDO pour un utilisateur IAM, veuillez consulter Activation d'une clé d'accès ou d'une clé de sécurité pour votre propre utilisateur IAM (console).

Si vous n'avez pas accès à une nouvelle clé d'accès ou une nouvelle clé de sécurité, vous pouvez activer un nouveau dispositif MFA virtuel ou jeton TOTP matériel. Pour plus d'informations, consultez l'un des liens suivants :

- Attribuez un périphérique MFA virtuel dans le Console de gestion AWS
- Attribuez un jeton TOTP matériel dans Console de gestion AWS

Configurations prises en charge pour l'utilisation des clés d'accès et des clés de sécurité

Vous pouvez utiliser les clés d'accès FIDO2 liées aux dispositifs, également appelées clés de sécurité, comme méthode d'authentification multifactorielle (MFA) avec IAM en utilisant les configurations actuellement prises en charge. Il s'agit notamment des dispositifs FIDO2 pris en charge par IAM et des navigateurs qui prennent en charge FIDO2.

Avant d'enregistrer votre appareil FIDO2, vérifiez que vous utilisez la dernière version du navigateur et du système d'exploitation (OS). Les fonctionnalités peuvent se comporter différemment selon les navigateurs, les authentificateurs et les clients du système d'exploitation. Si l'enregistrement de votre dispositif échoue sur un navigateur, vous pouvez essayer de vous enregistrer avec un autre navigateur.

FIDO2 est une norme d'authentification ouverte et une extension de FIDO U2F, offrant le même niveau de sécurité élevé basé sur la cryptographie à clé publique. FIDO2 comprend la spécification d'authentification Web (WebAuthn API) du W3C et le protocole FIDO Alliance (CTAP), un Client-to-Authenticator protocole de couche application. CTAP permet la communication entre le client ou la plateforme, comme un navigateur ou un système d'exploitation, avec un authentificateur externe.

Lorsque vous activez un authentificateur certifié FIDO dans AWS, la clé de sécurité crée une nouvelle paire de clés à utiliser uniquement. AWS Tout d'abord, saisissez vos informations d'identification. Lorsque vous y êtes invité, insérez la clé de sécurité, qui répond au défi d'authentification émis par AWS. Pour en savoir plus sur la norme FIDO2, veuillez consulter la rubrique Projet FIDO2.

Appareils FIDO2 pris en charge par AWS

IAM prend actuellement en charge les dispositifs de sécurité FIDO2 qui se connectent à vos dispositifs via USB, Bluetooth ou NFC. IAM prend également en charge les authentificateurs de plateforme comme TouchID ou FaceID. IAM ne prend pas en charge l'enregistrement de clés d'accès locales pour Windows Hello. Pour créer et utiliser des clés d'accès, les utilisateurs de Windows doivent utiliser l'authentification entre appareils, qui consiste à utiliser une clé d'accès provenant d'un appareil, comme un appareil mobile, ou une clé de sécurité matérielle pour se connecter sur un autre appareil, tel qu'un ordinateur portable.

> **Note**
>
> AWS nécessite l'accès au port USB physique de votre ordinateur pour vérifier votre appareil FIDO2. Les clés de sécurité ne fonctionnent pas avec une machine virtuelle, une connexion à distance ou le mode de navigation privée d'un navigateur.

La FIDO Alliance tient à jour une liste de tous les produits FIDO2 qui sont compatibles avec les spécifications FIDO.

Navigateurs prenant en charge FIDO2

La disponibilité des dispositifs de sécurité FIDO2 qui s'exécutent dans un navigateur Web dépend de la combinaison du navigateur et du système d'exploitation. Les navigateurs suivants prennent actuellement en charge l'utilisation de clés de sécurité :

Navigateur Web	macOS 10.15+	Windows 10	Linux	iOS 14.5+	Android 7+
Chrome	Oui	Oui	Oui	Oui	Non
Safari	Oui	Non	Non	Oui	Non
Edge	Oui	Oui	Non	Oui	Non

Navigateur Web	macOS 10.15+	Windows 10	Linux	iOS 14.5+	Android 7+
Firefox	Oui	Oui	Non	Oui	Non

> **Note**
>
> La plupart des versions de Firefox qui prennent en charge actuellement FIDO2 ne l'activent pas par défaut. Pour obtenir des instructions relatives à l'activation de FIDO2 dans Firefox, consultez Résolution des problèmes liés aux clés d'accès et aux clés de sécurité FIDO. Firefox sur macOS ne prend peut-être pas entièrement en charge les flux de travail d'authentification multi-appareils pour les clés d'accès. Il se peut que vous soyez invité à toucher une clé de sécurité au lieu de procéder à l'authentification multi-appareils. Nous vous recommandons d'utiliser un autre navigateur, tel que Chrome ou Safari, pour vous connecter à l'aide de clés d'accès sur macOS.

Pour plus d'informations sur la prise en charge des navigateurs pour un FIDO2-Certified appareil tel que YubiKey, consultez la section Prise en charge du système d'exploitation et du navigateur Web pour FIDO2 et U2F.

Plug-ins de navigateur

AWS ne prend en charge que les navigateurs qui prennent en charge FIDO2 de manière native. AWS ne prend pas en charge l'utilisation de plugins pour ajouter la prise en charge du navigateur FIDO2. Certains plug-ins de navigateur ne sont pas compatibles avec la norme FIDO2 et peuvent entraîner des résultats inattendus avec les clés de sécurité FIDO2.

Pour plus d'informations sur la désactivation des plug-ins et pour obtenir d'autres conseils de dépannage, consultez Je ne peux pas activer ma clé de sécurité FIDO.

Certifications des appareils

Nous ne saisissons et n'attribuons les certifications liées au dispositif, telles que la validation FIPS et le niveau de certification FIDO, que lors de l'enregistrement d'une clé de sécurité. La certification de votre appareil est extraite du FIDO Alliance Metadata Service (MDS).

Si le statut ou le niveau de certification de votre clé de sécurité change, celui-ci n'apparaîtra pas automatiquement dans les balises du dispositif. Pour mettre à jour les informations de certification

d'un appareil, enregistrez à nouveau l'appareil pour récupérer les informations de certification mises à jour.

AWS fournit les types de certification suivants comme clés de condition lors de l'enregistrement de l'appareil, obtenus à partir des niveaux de certification FIDO MDS : FIPS-140-2 FIPS-140-3, et FIDO. Vous pouvez restreindre l'enregistrement des authentificateurs dans les politiques IAM en fonction du type et du niveau de certification. Pour plus d'informations, consultez les politiques ci-dessous.

Exemples de politiques pour la certification des appareils

Les cas d'utilisation suivants présentent des exemples de politiques qui vous permettent d'enregistrer des appareils MFA avec des certifications FIPS.

Rubriques

- Cas d'utilisation 1 : autoriser l'enregistrement uniquement des appareils certifiés FIPS-140-2 L2
- Cas d'utilisation 2 : autoriser l'enregistrement des appareils certifiés FIPS-140-2 L2 et FIDO L1
- Cas d'utilisation 3 : autoriser l'enregistrement d'appareils dotés de certifications FIPS-140-2 L2 ou FIPS-140-3 L2
- Cas d'utilisation 4 : autoriser l'enregistrement des appareils certifiés FIPS-140-2 L2 et prenant en charge d'autres types d'authentification multifacteur, tels que les authentificateurs virtuels et le matériel TOTP

Cas d'utilisation 1 : autoriser l'enregistrement uniquement des appareils certifiés FIPS-140-2 L2

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
```

```
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        }
    ]
}
```

Cas d'utilisation 2 : autoriser l'enregistrement des appareils certifiés FIPS-140-2 L2 et FIDO L1

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2",
                    "iam:FIDO-certification": "L1"
                }
            }
        }
    ]
```

```
}
```

Cas d'utilisation 3 : autoriser l'enregistrement d'appareils dotés de certifications FIPS-140-2 L2 ou FIPS-140-3 L2

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L2"
                }
            }
        }
```

```
    ]
}
```

Cas d'utilisation 4 : autoriser l'enregistrement des appareils certifiés FIPS-140-2 L2 et prenant en charge d'autres types d'authentification multifacteur, tels que les authentificateurs virtuels et le matériel TOTP

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:EnableMFADevice",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:RegisterSecurityKey": "Create"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "iam:EnableMFADevice",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:RegisterSecurityKey": "Activate",
          "iam:FIDO-FIPS-140-2-certification": "L2"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "iam:EnableMFADevice",
      "Resource": "*",
      "Condition": {
        "Null": {
          "iam:RegisterSecurityKey": "true"
        }
```

```
      }
    }
  ]
}
```

AWS CLI and AWS API

AWS prend en charge l'utilisation de clés d'accès et de clés de sécurité uniquement dans le Console de gestion AWS. L'utilisation de clés d'accès et de clés de sécurité pour le MFA n'est pas prise en charge dans l'AWS API AWS CLI et, ni pour l'accès aux opérations de MFA-protected l'API.

Ressources supplémentaires

- Pour plus d'informations sur l'utilisation des clés d'accès et des clés de sécurité dans AWS, consultezAttribuez un mot de passe ou une clé de sécurité dans le Console de gestion AWS.
- Pour obtenir de l'aide sur la résolution des problèmes liés aux clés d'accès et aux clés de sécurité dans AWS, consultezRésolution des problèmes liés aux clés d'accès et aux clés de sécurité FIDO.
- Pour obtenir des informations générales sur le support FIDO2, consultez la rubrique Project FIDO2.

Attribuez un périphérique MFA virtuel dans le Console de gestion AWS

> **Important**
>
> AWS vous recommande d'utiliser un mot de passe ou une clé de sécurité pour le MFA AWS, dans la mesure du possible. Pour de plus amples informations, veuillez consulter Attribuez un mot de passe ou une clé de sécurité dans le Console de gestion AWS.

Vous pouvez utiliser un téléphone ou un autre appareil comme appareil d'authentification multifacteur (MFA) virtuel. Pour ce faire, installez une application mobile conforme à RFC 6238, un algorithme TOTP (mot de passe unique basé sur le temps) basé sur des normes. Ces applications génèrent un code d'authentification à six chiffres. Étant donné que les authentificateurs peuvent fonctionner sur des appareils mobiles non sécurisés et que les codes peuvent être partagés avec des parties non autorisées, le TOTP-based MFA n'offre pas le même niveau de sécurité que les options résistantes au phishing telles que les clés de sécurité et les clés d'accès FIDO2. https://en.wikipedia.org/wiki/FIDO_Alliance#FIDO2 Nous vous recommandons d'utiliser des clés d'accès ou des clés de sécurité pour le MFA afin de bénéficier d'une protection optimale contre les attaques telles que le phishing.

Si vous n'êtes pas encore en mesure d'utiliser des clés d'accès ou des clés de sécurité, nous vous recommandons d'utiliser un appareil MFA virtuel à titre de mesure provisoire en attendant l'approbation de l'achat de matériel ou l'arrivée de votre matériel.

La plupart des applications MFA virtuelles prennent en charge la création de plusieurs appareils virtuels, ce qui vous permet d'utiliser la même application pour plusieurs Comptes AWS utilisateurs. Vous pouvez enregistrer jusqu'à huit appareils MFA de n'importe quelle combinaison de types MFA auprès de vos utilisateurs Utilisateur racine d'un compte AWS et de ceux d'IAM. Vous n'avez besoin que d'un seul appareil MFA pour vous connecter Console de gestion AWS ou créer une session via le AWS CLI. Nous vous recommandons d'enregistrer plusieurs appareils MFA. Pour les applications d'authentification, nous recommandons également d'activer la fonctionnalité de sauvegarde ou de synchronisation dans le cloud afin d'éviter de perdre l'accès à votre compte en cas de perte ou de casse de votre dispositif.

AWS nécessite une application MFA virtuelle qui produit un OTP à six chiffres. Pour obtenir la liste des applications MFA virtuelles que vous pouvez utiliser, consultez la section Multi-Factor Authentification.

Rubriques

- Autorisations nécessaires
- Activation d'un dispositif MFA virtuel pour un utilisateur IAM (Console)
- Remplacer un périphérique MFA virtuel

Autorisations nécessaires

Pour gérer des dispositifs MFA virtuels pour votre utilisateur IAM, vous devez disposer des autorisations de la politique suivante : AWS : autorise les utilisateurs IAM authentifiés par MFA à gérer leur dispositif MFA sur la page Informations d'identification de sécurité.

Activation d'un dispositif MFA virtuel pour un utilisateur IAM (Console)

Vous pouvez utiliser IAM Console de gestion AWS pour activer et gérer un périphérique MFA virtuel pour un utilisateur IAM de votre compte. Vous pouvez attacher des balises à vos ressources IAM, y compris les appareils MFA virtuels, pour les identifier, les organiser et contrôler l'accès. Vous pouvez baliser des appareils MFA virtuels uniquement lorsque vous utilisez l' AWS API AWS CLI or. Pour activer et gérer un appareil MFA à l'aide de l' AWS API AWS CLI or, consultezAttribuez des appareils MFA dans l'API or AWS CLI AWS. Pour plus d'informations sur le balisage des ressources IAM, consultez Tags pour Gestion des identités et des accès AWS resources.

> **Note**
>
> Pour configurer l'authentification MFA, vous devez avoir accès physique au matériel sur lequel le dispositif MFA virtuel de l'utilisateur est hébergé. Par exemple, vous pouvez configurer le MFA pour un utilisateur qui utilisera un dispositif MFA virtuel s'exécutant sur un smartphone. Dans ce cas, vous devez avoir le smartphone à proximité afin de finaliser l'assistant. De ce fait, vous pouvez préférer laisser les utilisateurs configurer et gérer leurs propres dispositifs MFA virtuels. Dans ce cas, vous devez accorder aux utilisateurs l'autorisation d'exécuter les actions IAM nécessaires. Pour en savoir plus sur la politique IAM qui accorde ces autorisations et pour accéder à un exemple, veuillez consulter la rubrique Didacticiel IAM : permettre aux utilisateurs de gérer leurs informations d'identification et leurs paramètres MFA et la politique d'exemple AWS : autorise les utilisateurs IAM authentifiés par MFA à gérer leur dispositif MFA sur la page Informations d'identification de sécurité.

Pour activer un dispositif MFA virtuel pour un utilisateur IAM (console)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Dans la liste des Users (Utilisateurs), choisissez le nom d'utilisateur IAM.
4. Choisissez l'onglet Informations d'identification de sécurité. Sous Multi-factor Authentification (MFA), choisissez Attribuer un appareil MFA.
5. Dans l'assistant, saisissez un nom dans le champ Nom du dispositif, sélectionnez Application Authenticator, puis cliquez sur Suivant.

   IAM génère et affiche les informations de configuration du dispositif MFA virtuel, notamment un graphique de code QR. Le graphique est une représentation de la clé de configuration secrète que l'on peut saisir manuellement sur des périphériques qui ne prennent pas en charge les codes QR.
6. Ouvrez votre application MFA virtuelle. Pour obtenir la liste des applications que vous pouvez utiliser pour héberger des appareils MFA virtuels, consultez la section Multi-Factor Authentification.

   Si l'application MFA virtuelle prend en charge plusieurs comptes ou plusieurs dispositifs MFA virtuels, choisissez l'option permettant de créer un compte ou un dispositif MFA virtuel.

7. Déterminez si l'application MFA prend en charge les codes QR, puis effectuez l'une des actions suivantes :

   - Dans l'assistant, choisissez Show QR code (Afficher le code QR), puis utiliser l'application pour analyser le code QR. Il peut s'agir d'une icône de caméra ou d'une option Scanner le code qui utilise la caméra de l'appareil pour analyser le code.
   - Dans l'assistant, sélectionnez Show secret key (Afficher la clé secrète), puis saisissez la clé secrète dans votre application MFA.

   Une fois que vous avez terminé, le dispositif MFA virtuel commence à générer des mots de passe uniques.

8. Sur la page Configurer l'appareil, accédez à la zone Code MFA 1 et saisissez le mot de passe à usage unique affiché sur le dispositif MFA virtuel. Attendez jusqu'à 30 secondes pour que le dispositif génère un nouveau mot de passe unique. Saisissez ensuite le second mot de passe unique dans la zone MFA Code 2 (Code MFA 2). Choisissez Add MFA (Ajouter un dispositif MFA).

   > **Important**
   >
   > Envoyez votre demande immédiatement après avoir généré les codes. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, le dispositif MFA s'associe avec succès à l'utilisateur mais est désynchronisé. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période. Dans ce cas, vous pouvez resynchroniser le dispositif.

Le périphérique MFA virtuel est maintenant prêt à être utilisé avec AWS. Pour plus d'informations sur l'utilisation de l'authentification multifacteur avec le Console de gestion AWS, consultezConnexion compatible avec la MFA.

> **Note**
>
> Les appareils MFA virtuels non attribués dans votre Compte AWS compte sont supprimés lorsque vous ajoutez de nouveaux appareils MFA virtuels via Console de gestion AWS ou pendant le processus de connexion. Les dispositifs MFA virtuels non attribués sont des dispositifs présents dans votre compte, mais qui ne sont pas utilisés par l'utilisateur racine du compte ou les utilisateurs IAM pour le processus de connexion. Ils sont supprimés afin que

de nouveaux dispositifs MFA virtuels puissent être ajoutés à votre compte. Cela vous permet également de réutiliser les noms des appareils.

- Pour afficher les appareils MFA virtuels non attribués dans votre compte, vous pouvez utiliser la commande https://awscli.amazonaws.com/v2/documentation/api/latest/reference/iam/list-virtual-mfa-devices.html AWS CLI list-virtual-mfa-devices ou un appel d'API. https://docs.aws.amazon.com/IAM/latest/APIReference/API_ListVirtualMFADevices.html
- Pour désactiver un appareil MFA virtuel, vous pouvez utiliser la commande https://awscli.amazonaws.com/v2/documentation/api/latest/reference/iam/deactivate-mfa-device.html AWS CLI deactivate-mfa-device ou un appel d'API. https://docs.aws.amazon.com/IAM/latest/APIReference/API_DeactivateMFADevice.html Le dispositif ne sera plus attribué.
- Pour associer un périphérique MFA virtuel non attribué à votre utilisateur Compte AWS root ou à vos utilisateurs IAM, vous aurez besoin du code d'authentification généré par l'appareil ainsi que de la commande enable-mfa-device ou de l'appel AWS CLI d'API. https://docs.aws.amazon.com/IAM/latest/APIReference/API_EnableMFADevice.html

Remplacer un périphérique MFA virtuel

Vos utilisateurs Utilisateur racine d'un compte AWS et IAM peuvent enregistrer jusqu'à huit appareils MFA de n'importe quelle combinaison de types MFA. Si l'utilisateur perd un dispositif ou a besoin de le remplacer pour une raison quelconque, vous devez désactiver l'ancien dispositif. Vous pouvez alors ajouter le nouveau périphérique pour l'utilisateur.

- Pour désactiver le dispositif actuellement associé à un autre utilisateur IAM, consultez Désactivation d'un dispositif MFA.
- Pour ajouter un dispositif MFA virtuel de remplacement pour un autre utilisateur IAM, suivez les étapes de la procédure Activation d'un dispositif MFA virtuel pour un utilisateur IAM (Console) ci-dessus.
- Pour ajouter un périphérique MFA virtuel de remplacement pour le Utilisateur racine d'un compte AWS, suivez les étapes de la procédureActivation d'un dispositif MFA virtuel pour l'utilisateur racine (console).

Attribuez un jeton TOTP matériel dans Console de gestion AWS

> **Important**
>
> AWS vous recommande d'utiliser un mot de passe ou une clé de sécurité pour le MFA AWS, dans la mesure du possible. Pour de plus amples informations, veuillez consulter Attribuez un mot de passe ou une clé de sécurité dans le Console de gestion AWS.

Un jeton TOTP matériel génère un code numérique à six chiffres basé sur un algorithme TOTP (mot de passe unique à durée limitée). L'utilisateur doit saisir un code valide à partir du périphérique lorsqu'il y est invité lors de la connexion. Chaque dispositif MFA attribué à un utilisateur doit être unique ; un utilisateur ne peut pas saisir un code à partir du périphérique d'un autre utilisateur pour s'authentifier. Les dispositifs MFA ne peuvent pas être partagés entre plusieurs comptes ou plusieurs utilisateurs.

Les jetons TOTP matériels et les clés de sécurité FIDO sont des dispositifs physiques que vous achetez. Les appareils MFA matériels génèrent des codes TOTP pour l'authentification lorsque vous vous connectez à. AWS Ils utilisent des batteries, qui peuvent avoir besoin d'être remplacées et resynchronisées au fil du temps. AWS Les clés de sécurité FIDO, qui utilisent la cryptographie à clé publique, ne nécessitent pas de piles et offrent un processus d'authentification sans faille. Nous vous recommandons d'utiliser les clés de sécurité FIDO pour leur résistance au hameçonnage, ce qui constitue une alternative plus sûre aux appareils TOTP. De plus, les clés de sécurité FIDO peuvent prendre en charge plusieurs utilisateurs IAM ou utilisateurs racine sur le même appareil, ce qui améliore leur utilité pour la sécurité des comptes. Pour les spécifications et les informations d'achat pour les deux types d'appareils, voir Multi-Factor Authentification.

Vous pouvez activer un jeton TOTP matériel pour un utilisateur IAM à partir de la Console de gestion AWS ligne de commande ou de l'API IAM. Pour activer un périphérique MFA pour vous Utilisateur racine d'un compte AWS, consultezActivation d'un jeton TOTP matériel pour l'utilisateur root de l' (console).

Vous pouvez enregistrer jusqu'à huit appareils MFA de n'importe quelle combinaison des types MFA actuellement pris en charge auprès de vos utilisateurs Utilisateur racine d'un compte AWS et de ceux d'IAM. Avec plusieurs appareils MFA, vous n'avez besoin que d'un seul appareil MFA pour vous connecter Console de gestion AWS ou créer une session en AWS CLI tant qu'utilisateur.

> **Important**
>
> Nous vous recommandons d'activer plusieurs dispositifs MFA pour permettre à vos utilisateurs d'accéder en permanence à votre compte en cas de perte ou d'inaccessibilité d'un dispositif MFA.

> **Note**
>
> Si vous souhaitez activer le dispositif dans la ligne de commande, utilisez `aws iam enable-mfa-device` . Pour activer le dispositif MFA avec l'API IAM, utilisez l'opération `EnableMFADevice`.

Rubriques

- Autorisations nécessaires
- Activation d'un jeton TOTP matériel pour votre propre utilisateur IAM (console)
- Activation d'un jeton TOTP matériel pour un autre utilisateur IAM (console)
- Remplacer un périphérique MFA physique

Autorisations nécessaires

Pour gérer un jeton TOTP matériel pour votre propre utilisateur IAM tout en protégeant les MFA-related actions sensibles, vous devez disposer des autorisations prévues par la politique suivante :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
```

```
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Activation d'un jeton TOTP matériel pour votre propre utilisateur IAM (console)

Vous pouvez activer votre propre jeton TOTP matériel à partir de la Console de gestion AWS.

> **Note**
>
> Avant d'activer un jeton TOTP matériel, vous devez y avoir accès physiquement.

Pour activer un jeton TOTP matériel pour votre propre utilisateur IAM (console)

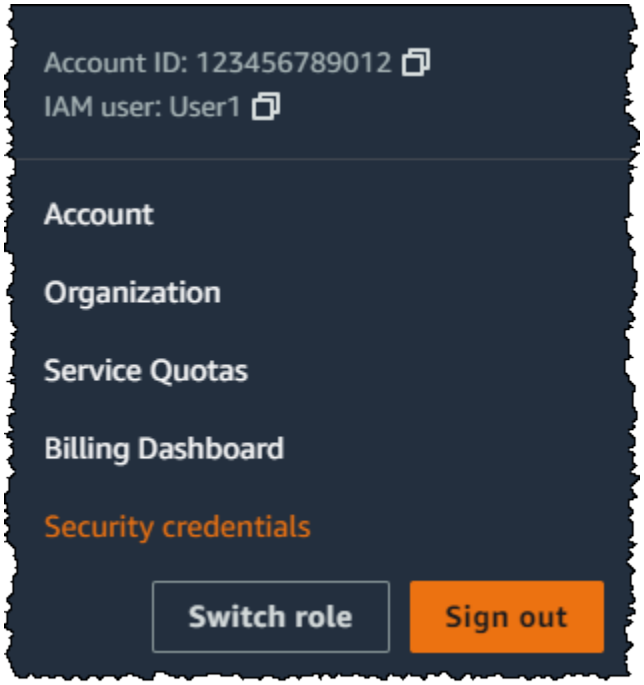
1. Utilisez votre identifiant de AWS compte ou votre alias de compte, votre nom d'utilisateur IAM et votre mot de passe pour vous connecter à la console IAM.

   > **Note**
   >
   > Pour votre commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser votre nom d'utilisateur IAM et les informations de votre compte. Si vous vous

êtes déjà connecté en tant qu'utilisateur différent, sélectionnez Sign in to a different account (Se connecter à un compte différent) en bas de la page pour revenir à la page de connexion principale. À partir de là, vous pouvez saisir votre identifiant de AWS compte ou votre alias de compte pour être redirigé vers la page de connexion utilisateur IAM de votre compte.

Pour obtenir votre Compte AWS identifiant, contactez votre administrateur.

2. Dans la barre de navigation, en haut à droite, choisissez votre nom d'utilisateur, puis Security credentials (Informations d'identification de sécurité).



3. Dans l'onglet Informations d'identification AWS IAM, dans la section Multi-factor Authentification (MFA), choisissez Attribuer un appareil MFA.
4. Dans l'assistant, tapez le nom du dispositif, choisissez Hardware TOTP token (Jeton TOTP matériel), puis Next (Suivant).
5. Saisissez le numéro de série du périphérique. Le numéro de série se situe généralement l'arrière du périphérique.
6. Dans la zone MFA code 1, saisissez le code à six chiffres qui s'affichent sur le dispositif MFA. Vous devrez peut-être appuyer sur le bouton situé à l'avant du périphérique pour afficher le numéro.



7. Attendez 30 secondes que le périphérique actualise le code, puis saisissez la nouvelle série de six chiffres dans la zone MFA code 2. Vous devrez peut-être appuyer à nouveau sur le bouton situé à l'avant du périphérique pour afficher le second numéro.
8. Choisissez Add MFA (Ajouter un dispositif MFA).

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes d'authentification. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, l'dispositif MFA s'associe avec succès à l'utilisateur mais se désynchronise. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période. Dans ce cas, vous pouvez resynchroniser le dispositif.

L'appareil est prêt à être utilisé avec AWS. Pour plus d'informations sur l'utilisation de l'authentification MFA avec l'interface Console de gestion AWS, veuillez consulter Connexion compatible avec la MFA.

Activation d'un jeton TOTP matériel pour un autre utilisateur IAM (console)

Vous pouvez activer un jeton TOTP matériel pour un autre utilisateur IAM à partir de la Console de gestion AWS.

Pour activer un jeton TOTP matériel pour un autre utilisateur IAM (console)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom de l'utilisateur pour lequel vous souhaitez activer l'authentification MFA.
4. Choisissez l'onglet Informations d'identification de sécurité. Sous Multi-factor Authentification (MFA), choisissez Attribuer un appareil MFA.
5. Dans l'assistant, tapez le nom du dispositif, choisissez Hardware TOTP token (Jeton TOTP matériel), puis Next (Suivant).

6. Saisissez le numéro de série du périphérique. Le numéro de série se situe généralement l'arrière du périphérique.
7. Dans la zone MFA code 1, saisissez le code à six chiffres qui s'affichent sur le dispositif MFA. Vous devrez peut-être appuyer sur le bouton situé à l'avant du périphérique pour afficher le numéro.



8. Attendez 30 secondes que le périphérique actualise le code, puis saisissez la nouvelle série de six chiffres dans la zone MFA code 2. Vous devrez peut-être appuyer à nouveau sur le bouton situé à l'avant du périphérique pour afficher le second numéro.
9. Choisissez Add MFA (Ajouter un dispositif MFA).

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes d'authentification. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, l'dispositif MFA s'associe avec succès à l'utilisateur mais se désynchronise. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période. Dans ce cas, vous pouvez resynchroniser le dispositif.

L'appareil est prêt à être utilisé avec AWS. Pour plus d'informations sur l'utilisation de l'authentification MFA avec l'interface Console de gestion AWS, veuillez consulter Connexion compatible avec la MFA.

Remplacer un périphérique MFA physique

Jusqu'à huit appareils MFA, quelle que soit la combinaison des types MFA actuellement pris en charge, peuvent être attribués à un utilisateur à la fois avec vous Utilisateur racine d'un compte AWS et vos utilisateurs IAM. Si l'utilisateur perd un périphérique ou a besoin de le remplacer pour une raison quelconque, vous devez désactiver l'ancien périphérique. Vous pouvez alors ajouter le nouveau périphérique pour l'utilisateur.

- Pour désactiver le périphérique actuellement associé à un utilisateur, consultez la page Désactivation d'un dispositif MFA.

- Pour ajouter un jeton TOTP matériel de remplacement pour un utilisateur IAM, suivez les étapes de la procédure Activation d'un jeton TOTP matériel pour un autre utilisateur IAM (console) plus haut dans cette rubrique.
- Pour ajouter un jeton matériel TOTP de remplacement pour le Utilisateur racine d'un compte AWS, suivez les étapes de la procédure décrite Activation d'un jeton TOTP matériel pour l'utilisateur root de l' (console) plus haut dans cette rubrique.

Attribuez des appareils MFA dans l'API or AWS CLI AWS

Vous pouvez utiliser des AWS CLI commandes ou des opérations d' AWS API pour activer un dispositif MFA virtuel pour un utilisateur IAM. Vous ne pouvez pas activer un périphérique MFA Utilisateur racine d'un compte AWS avec l' AWS API AWS CLI, Tools for Windows PowerShell ou tout autre outil de ligne de commande. Toutefois, vous pouvez utiliser le Console de gestion AWS pour activer un périphérique MFA pour l'utilisateur root.

Lorsque vous activez un dispositif MFA à partir du Console de gestion AWS, la console exécute plusieurs étapes pour vous. Si vous créez plutôt un appareil virtuel à l' AWS CLI aide des outils pour Windows PowerShell ou de AWS l'API, vous devez effectuer les étapes manuellement et dans le bon ordre. Par exemple, pour créer un dispositif MFA virtuel, vous devez créer l'objet IAM et extraire le code sous forme de chaîne ou de graphique de code QR. Ensuite, vous devez synchroniser le périphérique et l'associer à un utilisateur IAM. Consultez la section Exemples de New- IAMVirtual MFADevice pour plus de détails. Dans le cas d'un périphérique physique, vous ignorez l'étape de création et passez directement à la synchronisation du périphérique et l'association à un utilisateur.

Vous pouvez attacher des balises à vos ressources IAM, y compris les appareils MFA virtuels, pour les identifier, les organiser et contrôler l'accès. Vous pouvez étiqueter les appareils MFA virtuels uniquement lorsque vous utilisez l'API AWS CLI or AWS .

Un utilisateur IAM utilisant le kit SDK ou l'interface de ligne de commande peut activer un dispositif MFA supplémentaire en appelant `EnableMFADevice` ou désactiver un dispositif MFA existant en appelant `DeactivateMFADevice`. Pour y parvenir, il doit d'abord appeler `GetSessionToken` et soumettre des codes MFA avec un dispositif MFA existant. Cet appel renvoie des informations d'identification de sécurité temporaires qui peuvent ensuite être utilisées pour signer des opérations d'API nécessitant une authentification MFA. Pour un exemple de demande et de réponse, consultez `GetSessionToken` : informations d'identification temporaires pour les utilisateurs qui se trouvent dans des environnements non fiables.

Pour créer l'entité de périphérique virtuel dans IAM pour représenter un dispositif MFA virtuel

Ces commandes fournissent un ARN pour le périphérique qui est utilisé à la place du numéro de série dans un grand nombre des commandes suivantes.

- AWS CLI: `aws iam create-virtual-mfa-device`
- AWS API : `CreateVirtualMFADevice`

Pour activer un périphérique MFA à utiliser avec AWS

Ces commandes synchronisent l'appareil avec un utilisateur AWS et l'associent à celui-ci. S'il s'agit d'un périphérique virtuel, utilisez son ARN en tant que numéro de série.

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes d'authentification. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, l'dispositif MFA s'associe avec succès à l'utilisateur mais se désynchronise. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période. Dans ce cas, vous pouvez resynchroniser l'appareil à l'aide des commandes décrites ci-dessous.

- AWS CLI: `aws iam enable-mfa-device`
- AWS API : `EnableMFADevice`

Pour désactiver un périphérique

Utilisez ces commandes pour dissocier le périphérique de l'utilisateur et le désactiver. S'il s'agit d'un périphérique virtuel, utilisez son ARN en tant que numéro de série. Vous devez également supprimer l'entité de périphérique virtuel séparément.

- AWS CLI: `aws iam deactivate-mfa-device`
- AWS API : `DeactivateMFADevice`

Pour afficher la liste des entités de dispositifs MFA virtuels

Utilisez ces commandes pour afficher la liste des entités de dispositifs MFA virtuels.

- AWS CLI: `aws iam list-virtual-mfa-devices`

- AWS API : `ListVirtualMFADevices`

Pour baliser un appareil MFA virtuel

Utilisez ces commandes pour baliser un appareil MFA virtuel.

- AWS CLI: `aws iam tag-mfa-device`
- AWS API : `TagMFADevice`

Pour répertorier les balises d'un appareil MFA virtuel

Utilisez ces commandes pour répertorier les balises attachées à un appareil MFA virtuel.

- AWS CLI: `aws iam list-mfa-device-tags`
- AWS API : `ListMFADeviceTags`

Pour supprimer la balise d'un appareil MFA virtuel

Utilisez ces commandes pour supprimer les balises attachées à un appareil MFA virtuel.

- AWS CLI: `aws iam untag-mfa-device`
- AWS API : `UntagMFADevice`

Pour resynchroniser un dispositif MFA

Utilisez ces commandes si le périphérique génère des codes qui ne sont pas acceptés par AWS. S'il s'agit d'un périphérique virtuel, utilisez son ARN en tant que numéro de série.

- AWS CLI: `aws iam resync-mfa-device`
- AWS API : `ResyncMFADevice`

Pour supprimer une entité de dispositif MFA virtuel dans IAM

Après avoir dissocié le périphérique de l'utilisateur, vous pouvez supprimer l'entité de périphérique.

- AWS CLI: `aws iam delete-virtual-mfa-device`
- AWS API : `DeleteVirtualMFADevice`

Pour récupérer un dispositif MFA virtuel qui est perdu ou ne fonctionne pas

Parfois, l'appareil d'un utilisateur qui héberge l'application MFA virtuelle est perdu, remplacé ou ne fonctionne pas. Dans ce cas, l'utilisateur ne peut pas le récupérer par lui-même. L'utilisateur doit contacter un administrateur pour désactiver le dispositif. Pour de plus amples informations, veuillez consulter Restauration d'une identité protégée par MFA dans IAM.

Vérification du statut MFA

Utilisez la console IAM pour vérifier si un périphérique MFA valide est activé pour un utilisateur Utilisateur racine d'un compte AWS ou un utilisateur IAM.

Pour vérifier le statut de l'authentification MFA d'un utilisateur racine

1. Connectez-vous à l' Console de gestion AWS aide de vos informations d'identification d'utilisateur root, puis ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans la barre de navigation, en haut à droite, choisissez votre nom d'utilisateur, puis Security credentials (Informations d'identification de sécurité).
3. Vérifiez sous Multi-factor Authentification (MFA) si l'authentification MFA est activée ou désactivée. Si la MFA n'a pas été activée, un symbole d'alerte ( ) est affiché.

Si vous souhaitez activer l'authentification MFA pour le compte, consultez l'une des sections suivantes :

- Activation d'un dispositif MFA virtuel pour l'utilisateur racine (console)
- Activation d'une clé d'accès ou d'une clé de sécurité pour l'utilisateur racine (console)
- Activation d'un jeton TOTP matériel pour l'utilisateur root de l' (console)

Pour vérifier le statut de l'authentification MFA d'utilisateurs IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Au besoin, ajoutez la colonne MFA à la table des utilisateurs en procédant comme suit :

a. Au-dessus de la table à l'extrême droite, choisissez l'icône des paramètres ( ).

b. Dans Manage Columns (Gérer les colonnes), sélectionnez MFA.

c. (Facultatif) Décochez la case des en-têtes de colonne que vous ne souhaitez pas voir apparaître dans le tableau des utilisateurs.

d. Choisissez Fermer pour revenir à la liste des utilisateurs.

4. La colonne MFA vous fournit des informations sur le dispositif MFA qui est activé. Si aucun dispositif MFA n'est actif pour l'utilisateur, la console affiche None (Aucun). Si l'utilisateur dispose d'un dispositif MFA activé, la colonne MFA affiche le type de dispositif activé avec la valeur de Virtuel, Clé de sécurité, Matériel ou SMS.

> **Note**
>
> AWS fin du support pour l'activation de l'authentification multifactorielle par SMS (MFA). Nous recommandons aux clients dont les utilisateurs IAM utilisent un dispositif MFA basé sur les SMS de passer à l'une des méthodes alternatives suivantes : dispositif MFA virtuel (logiciel), clé de sécurité FIDO ou dispositif MFA matériel. Vous pouvez identifier les utilisateurs dans votre compte avec un dispositif MFA SMS affecté. Pour ce faire, accédez à la console IAM, choisissez Utilisateurs dans le panneau de navigation, puis recherchez les utilisateurs avec SMS mentionné dans la colonne MFA de la table.

5. Pour afficher des informations supplémentaires sur le dispositif MFA pour un utilisateur, choisissez le nom de l'utilisateur dont vous souhaitez vérifier l'état MFA. Choisissez ensuite l'onglet Informations d'identification de sécurité.

6. Si aucun dispositif MFA n'est actif pour l'utilisateur, la console affiche Aucun dispositif MFA. Attribuez un dispositif MFA pour améliorer la sécurité de votre AWS environnement dans la section Multi-factor Authentification (MFA). Si les appareils MFA sont activés pour l'utilisateur, la section Multi-factor d'authentification (MFA) affiche des détails sur les appareils :

   - Nom du dispositif
   - Type du dispositif
   - L'identifiant du périphérique, tel que le numéro de série d'un périphérique physique ou l'ARN AWS d'un périphérique virtuel
   - Quand le dispositif a été créé

Pour supprimer ou resynchroniser un dispositif, cliquez sur le bouton radio à côté du dispositif, puis choisissez Remove (Supprimer) ou Resync (Resynchroniser).

Pour plus d'informations sur l'activation de l'authentification MFA, consultez la documentation suivante :

- Attribuez un périphérique MFA virtuel dans le Console de gestion AWS
- Attribuez un mot de passe ou une clé de sécurité dans le Console de gestion AWS
- Attribuez un jeton TOTP matériel dans Console de gestion AWS

Resynchronisation des dispositifs MFA virtuels et matériels

Vous pouvez l'utiliser AWS pour resynchroniser vos appareils d'authentification multifacteur (MFA) virtuels et matériels. Si votre dispositif n'est pas synchronisé lorsque vous essayez de l'utiliser, la tentative de connexion échoue et IAM vous invite à resynchroniser le dispositif.

> **Note**
>
> Les clés de sécurité FIDO ne se désynchronisent pas. Si une clé de sécurité FIDO est perdue ou endommagée, vous pouvez la désactiver. Pour plus d'informations sur la désactivation de tout type de dispositif MFA, consultez Pour désactiver un dispositif MFA pour un autre utilisateur IAM (console).

En tant qu' AWS administrateur, vous pouvez resynchroniser les appareils MFA virtuels et matériels de vos utilisateurs IAM s'ils ne sont pas synchronisés.

Si votre appareil Utilisateur racine d'un compte AWS MFA ne fonctionne pas, vous pouvez le resynchroniser à l'aide de la console IAM avec ou sans terminer le processus de connexion. Si vous ne parvenez pas à resynchroniser votre appareil, vous devrez peut-être le désassocier et le réassocier. Pour en savoir plus à ce sujet, veuillez consulter les rubriques Désactivation d'un dispositif MFA et AWS Multi-factor authentification dans IAM.

Rubriques

- Autorisations nécessaires
- Resynchronisation des dispositifs MFA matériels et virtuels (console IAM)
- Resynchronisation des périphériques MFA virtuels et matériels (AWS CLI)

- Resynchronisation des périphériques MFA virtuels et matériels (AWS API)

Autorisations nécessaires

Pour resynchroniser des dispositifs MFA virtuels ou matériels pour votre propre utilisateur IAM, vous devez disposer des autorisations de la politique suivante. Cette politique ne vous permet pas de créer ou de désactiver un dispositif.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToViewAndManageTheirOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "BlockAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
```

```
                }
            }
        }
    ]
}
```

Resynchronisation des dispositifs MFA matériels et virtuels (console IAM)

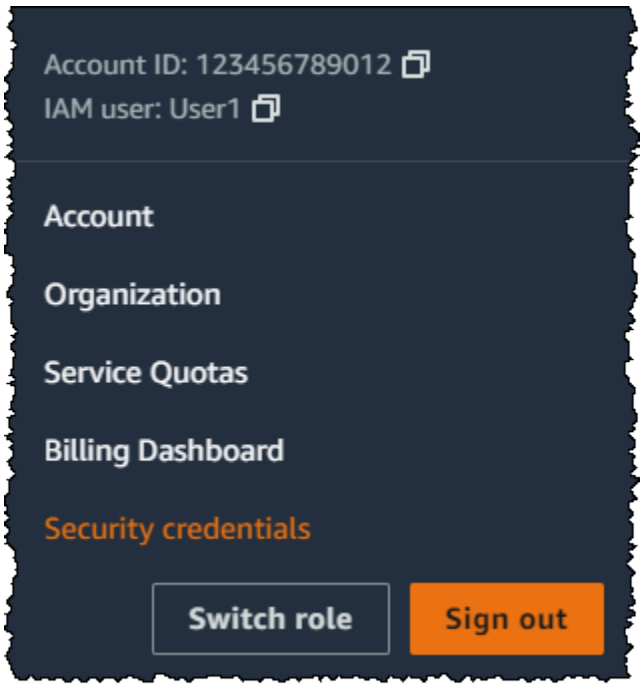
Vous pouvez utiliser la console IAM pour resynchroniser les dispositifs virtuels du matériel MFA.

Pour resynchroniser un dispositif MFA matériel ou virtuel pour votre propre utilisateur IAM (console)

1. Utilisez votre identifiant de AWS compte ou votre alias de compte, votre nom d'utilisateur IAM et votre mot de passe pour vous connecter à la console IAM.

   > **Note**
   >
   > Pour votre commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser votre nom d'utilisateur IAM et les informations de votre compte. Si vous vous êtes déjà connecté en tant qu'utilisateur différent, sélectionnez Sign in to a different account (Se connecter à un compte différent) en bas de la page pour revenir à la page de connexion principale. À partir de là, vous pouvez saisir votre identifiant de AWS compte ou votre alias de compte pour être redirigé vers la page de connexion utilisateur IAM de votre compte.

   Pour obtenir votre Compte AWS identifiant, contactez votre administrateur.
2. Dans la barre de navigation, en haut à droite, choisissez votre nom d'utilisateur, puis Security credentials (Informations d'identification de sécurité).



3. Dans l'onglet Informations d'identification AWS IAM, dans la section Multi-factor Authentification (MFA), cliquez sur le bouton radio à côté du périphérique MFA et choisissez Resynchroniser.
4. Entrez les deux prochains codes générés séquentiellement à partir du périphérique dans les champs MFA code 1 et MFA code 2. Puis choisissez Resync (Resynchroniser).

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, cette dernière semble fonctionner mais l'appareil reste désynchronisé. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période.

Pour resynchroniser un dispositif MFA matériel ou virtuel pour un autre utilisateur IAM (console)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation, choisissez Utilisateurs, puis sélectionnez le nom de l'utilisateur dont le dispositif MFA doit être resynchronisé.

3. Choisissez l'onglet Informations d'identification de sécurité. Dans la section Multi-factor Authentification (MFA), cliquez sur le bouton radio à côté du périphérique MFA et choisissez Resynchroniser.
4. Entrez les deux prochains codes générés séquentiellement à partir du périphérique dans les champs MFA code 1 et MFA code 2. Puis choisissez Resync (Resynchroniser).

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, cette dernière semble fonctionner mais l'appareil reste désynchronisé. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période.

Pour resynchroniser votre dispositif MFA d'utilisateur racine avant la connexion (console)

1. Sur la page Amazon Web Services Sign In With Authentication Device (connexion à Amazon Web Services à l'aide de MFA), choisissez Having problems with your authentication device? (des problèmes avec votre dispositif d'authentification ?) Click here (Cliquez ici).

> **Note**
>
> Il se peut que vous remarquiez des textes différents, tels que se connecter à l'aide de MFA et dépanner votre dispositif d'authentification. Toutefois, les mêmes fonctions sont fournies.

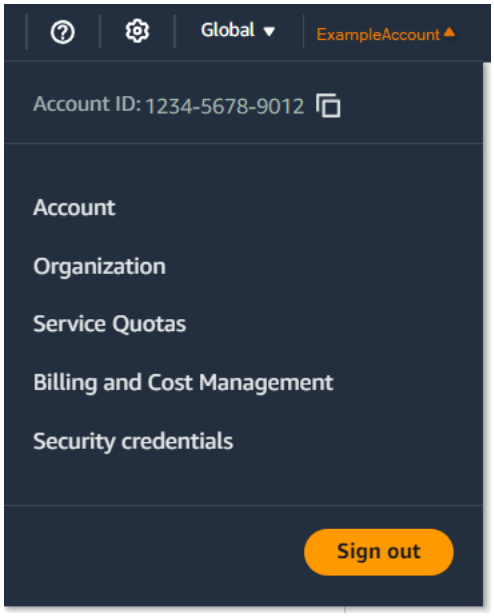
2. Dans la section Re-Sync Avec nos serveurs, saisissez les deux prochains codes générés séquentiellement par l'appareil dans le code MFA 1 et le code MFA 2. Choisissez ensuite le dispositif Re-sync d'authentification.
3. Si besoin, saisissez à nouveau votre mot de passe et choisissez Sign in (Connexion). Ensuite, procédez à la connexion à l'aide de votre dispositif MFA.

Pour resynchroniser votre dispositif MFA d'utilisateur racine après la connexion (console)

1. Connectez-vous à la console IAM en tant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   > **Note**
   >
   > En tant qu'utilisateur root, vous ne pouvez pas vous connecter à la page Se connecter en tant qu'utilisateur IAM. Si la page Se connecter en tant qu'utilisateur IAM s'affiche, choisissez Se connecter à l'aide de l'adresse e-mail de l'utilisateur root en bas de la page. Pour obtenir de l'aide pour vous connecter en tant qu'utilisateur root, consultez la section Connexion en Console de gestion AWS tant qu'utilisateur root dans le Guide de Connexion à AWS l'utilisateur.

2. À droite de la barre de navigation, sélectionnez le nom de votre compte, puis Security Credentials (Informations d'identification de sécurité). Au besoin, choisissez Continue to Security credentials (Passer aux informations d'identification de sécurité).



3. Développez la section Multi-factor Authentification (MFA) de la page.
4. Cliquez sur le bouton radio en regard du dispositif et choisissez Resync (Resynchroniser).
5. Dans la boîte de dialogue Resync MFA device (Resynchroniser le dispositif MFA), entrez les deux prochains codes générés séquentiellement à partir du dispositif dans les champs MFA code 1 et MFA code 2. Puis choisissez Resync (Resynchroniser).

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes. Si vous générez les codes, puis attendez trop longtemps avant d'envoyer la demande, le dispositif MFA s'associe avec succès à l'utilisateur mais est désynchronisé. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période.

Resynchronisation des périphériques MFA virtuels et matériels (AWS CLI)

Vous pouvez resynchroniser les dispositifs MFA virtuels et matériels à partir de l'interface AWS CLI.

Pour resynchroniser un périphérique MFA virtuel ou matériel pour un utilisateur IAM (AWS CLI)

En réponse à une invite de commande, émettez la commande aws iam resync-mfa-device :

- Dispositif MFA virtuel : spécifiez l'Amazon Resource Name (ARN) du périphérique en tant que numéro de série.

```
aws iam resync-mfa-device --user-name Richard --serial-number
 arn:aws:iam::123456789012:mfa/RichardsMFA --authentication-code1 123456 --
authentication-code2 987654
```

- Dispositif MFA matériel : spécifiez le numéro de série du périphérique matériel en tant que numéro de série. Le format est spécifique au fournisseur. Par exemple, vous pouvez acheter un jeton gemalto auprès d'Amazon. Son numéro de série est généralement composé de quatre lettres suivies de quatre chiffres.

```
aws iam resync-mfa-device --user-name Richard --serial-number ABCD12345678 --
authentication-code1 123456 --authentication-code2 987654
```

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, cette dernière échoue car les codes expirent après une courte période.

Resynchronisation des périphériques MFA virtuels et matériels (AWS API)

IAM dispose d'un appel d'API qui effectue la synchronisation. Dans ce cas, nous vous recommandons d'accorder à vos utilisateurs de dispositifs MFA virtuels et matériels l'autorisation d'accès à cet appel d'API. Créez ensuite un outil basé sur cet appel d'API afin que vos utilisateurs puissent resynchroniser leurs périphériques chaque fois que cela est nécessaire.

Pour resynchroniser un périphérique MFA virtuel ou matériel pour un utilisateur IAM (AWS API)

- Envoyez la demande ResyncMFADevice.

Désactivation d'un dispositif MFA

Si vous rencontrez des difficultés pour vous connecter avec un dispositif d'authentification multifactorielle (MFA) en tant qu'utilisateur IAM, contactez votre administrateur pour obtenir de l'aide.

En tant qu'administrateur, vous pouvez désactiver le dispositif pour un autre utilisateur IAM. Cela permet à l'utilisateur de se connecter sans utiliser MFA. Vous pouvez faire ceci comme solution provisoire en attendant que le dispositif MFA soit remplacé ou si le périphérique est indisponible temporairement. Par contre, nous vous recommandons d'activer un nouveau périphérique pour l'utilisateur dès que possible. Pour savoir comment activer un nouveau dispositif MFA, consultez AWS Multi-factor authentification dans IAM.

> **Note**
>
> Si vous utilisez l'API ou si vous AWS CLI souhaitez supprimer un utilisateur Compte AWS, vous devez désactiver ou supprimer le dispositif MFA de l'utilisateur. Vous effectuez cette modification dans le cadre du processus de suppression de l'utilisateur. Pour en savoir plus sur la suppression d'utilisateurs, consultez Suppression ou désactivation d'un utilisateur IAM.

Rubriques

- Désactivation des dispositifs MFA (console)
- Désactivation des appareils MFA (AWS CLI)
- Désactivation des appareils MFA (AWS API)

Désactivation des dispositifs MFA (console)

Pour désactiver un dispositif MFA pour un autre utilisateur IAM (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Pour désactiver le dispositif MFA pour un utilisateur, choisissez le nom de l'utilisateur dont vous souhaitez supprimer le MFA.
4. Choisissez l'onglet Informations d'identification de sécurité.
5. Sous Multi-factor Authentification (MFA), cliquez sur le bouton radio situé à côté du périphérique MFA, choisissez Supprimer, puis Supprimer.

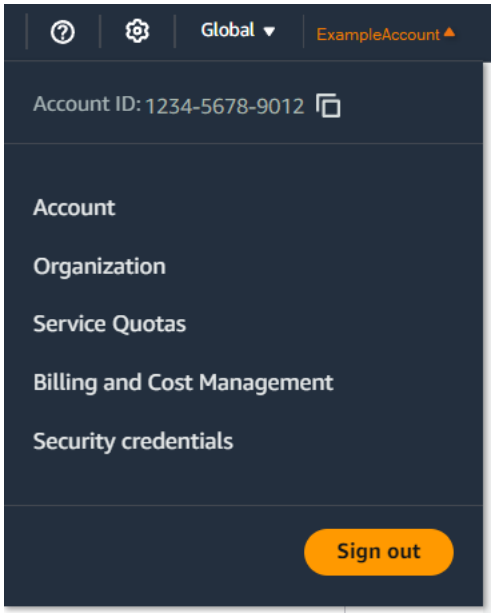
   L'appareil est retiré de AWS. Il ne peut pas être utilisé pour se connecter ou authentifier des demandes tant qu'il n'est pas réactivé et associé à un AWS utilisateur ou. Utilisateur racine d'un compte AWS

Pour désactiver le dispositif MFA de votre Utilisateur racine d'un compte AWS (console)

1. Connectez-vous à la console IAM en tant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   > **Note**
   >
   > En tant qu'utilisateur root, vous ne pouvez pas vous connecter à la page Se connecter en tant qu'utilisateur IAM. Si la page Se connecter en tant qu'utilisateur IAM s'affiche, choisissez Se connecter à l'aide de l'adresse e-mail de l'utilisateur root en bas de la page. Pour obtenir de l'aide pour vous connecter en tant qu'utilisateur root, consultez la section Connexion en Console de gestion AWS tant qu'utilisateur root dans le Guide de Connexion à AWS l'utilisateur.

2. À droite de la barre de navigation, sélectionnez le nom de votre compte, puis Security Credentials (Informations d'identification de sécurité). Au besoin, choisissez Continue to Security credentials (Passer aux informations d'identification de sécurité).



3. Dans la section Multi-factor Authentification (MFA), cliquez sur le bouton radio situé à côté du dispositif MFA que vous souhaitez désactiver, puis sélectionnez Supprimer.
4. Cliquez sur Supprimer.

   Le dispositif MFA est désactivé pour l' Compte AWS. Vérifiez que l'e-mail qui vous est associé contient Compte AWS un message de confirmation provenant d'Amazon Web Services. L'e-mail vous informe que votre authentification multi-facteur (MFA) Amazon Web Services a été désactivée. Le message viendra de `@amazon.com` ou `@aws.amazon.com`.

> **Note**
>
> Les appareils MFA virtuels non assignés sont supprimés lorsque vous ajoutez de nouveaux appareils MFA virtuels, que ce soit pendant ou pendant Console de gestion AWS le processus de connexion. Compte AWS Les dispositifs MFA virtuels non attribués sont des dispositifs présents dans votre compte, mais qui ne sont pas utilisés par l'utilisateur racine du compte ou les utilisateurs IAM pour le processus de connexion. Ils sont supprimés afin que de nouveaux dispositifs MFA virtuels puissent être ajoutés à votre compte. Cela vous permet également de réutiliser les noms des appareils.

Désactivation des appareils MFA (AWS CLI)

Pour désactiver un dispositif MFA pour un utilisateur IAM (AWS CLI)

- Exécutez cette commande : `aws iam deactivate-mfa-device`

Désactivation des appareils MFA (AWS API)

Pour désactiver un dispositif MFA pour un utilisateur IAM (AWS API)

- Appelez cette opération : `DeactivateMFADevice`

Restauration d'une identité protégée par MFA dans IAM

Si votre périphérique MFA virtuel ou votre jeton TOTP matériel semble fonctionner correctement, mais que vous ne pouvez pas l'utiliser pour accéder à vos AWS ressources, il est possible qu'il ne soit pas synchronisé avec. AWS Pour plus d'informations sur la synchronisation d'un dispositif MFA virtuel ou d'un dispositif MFA matériel, consultez Resynchronisation des dispositifs MFA virtuels et matériels. Les clés de sécurité FIDO ne se désynchronisent pas.

Si l'appareil MFA d'un utilisateur Utilisateur racine d'un compte AWS est perdu, endommagé ou ne fonctionne pas, vous pouvez récupérer l'accès à votre compte. Les utilisateurs IAM doivent contacter un administrateur pour désactiver le périphérique.

> **Important**
>
> Nous vous recommandons d'activer plusieurs dispositifs MFA. L'enregistrement de plusieurs dispositifs MFA permet de garantir la continuité de l'accès en cas de perte ou de casse d'un dispositif. Vos utilisateurs Utilisateur racine d'un compte AWS et ceux d'IAM peuvent enregistrer jusqu'à huit appareils MFA de tout type.

Prérequis – Utilisation d'un autre appareil MFA

Si le dispositif d'authentification multifactorielle (MFA) est perdu, endommagé ou défaillant, vous pouvez vous connecter à l'aide d'un autre dispositif MFA enregistré au même utilisateur racine ou utilisateur IAM.

Pour vous connecter à l'aide d'un autre appareil MFA

1. Connectez-vous à l'Console de gestion AWS à l'aide de votre ID d' Compte AWS ou de votre alias de compte et de votre mot de passe.
2. Sur la page Vérification supplémentaire requise ou sur la page Multi-factor d'authentification, choisissez Essayez une autre méthode MFA.
3. Authentifiez-vous avec le type de dispositif MFA que vous avez sélectionné.
4. L'étape suivante varie selon que vous vous êtes connecté avec succès avec un autre dispositif MFA.
   - Si vous vous êtes connecté avec succès, vous pouvez Resynchronisation des dispositifs MFA virtuels et matériels, ce qui peut résoudre le problème. Si votre dispositif MFA est perdu ou endommagé, vous pouvez le désactiver. Pour plus d'informations sur la désactivation de tout type de dispositif MFA, consultez Désactivation d'un dispositif MFA.
   - Si vous ne parvenez pas à vous connecter avec la MFA, suivez les étapes indiquées dans Récupération d'un dispositif MFA d'utilisateur racine ou Récupération d'un dispositif MFA d'utilisateur IAM pour récupérer votre identité protégée par la MFA.

Récupération d'un dispositif MFA d'utilisateur racine

Si vous ne pouvez pas vous connecter par MFA, vous pouvez utiliser d'autres méthodes d'authentification pour vous connecter en vérifiant votre identité à l'aide de l'e-mail et du numéro de téléphone de contact principal enregistrés avec votre compte.

Assurez-vous de pouvoir accéder à l'e-mail et au numéro de téléphone du contact principal associés à votre compte avant d'utiliser d'autres facteurs d'authentification pour vous connecter en tant qu'utilisateur racine. Si vous devez mettre à jour le numéro de téléphone du contact principal, connectez-vous en tant qu'utilisateur IAM avec un accès Administrateur au lieu de l'utilisateur racine. Pour obtenir des instructions supplémentaires sur la mise à jour des informations de contact du compte, consultez la section Modification de vos informations de contact dans le Guide de l'utilisateur AWS Billing . Si vous n'avez pas accès à une adresse e-mail et à un numéro de téléphone de contact principal, vous devez contacter AWS Support.

> **Important**
>
> Nous vous recommandons de garder à jour l'adresse e-mail et le numéro de téléphone de contact associés à votre utilisateur root pour une restauration réussie du compte. Pour plus

d'informations, consultez Mettre à jour le contact principal pour votre Compte AWS dans le guide de référence d'Gestion de compte AWS .

Pour vous connecter en utilisant d'autres facteurs d'authentification en tant que Utilisateur racine d'un compte AWS

1. Connectez-vous en Console de gestion AWS tant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.
2. Sur la page Vérification supplémentaire nécessaire, choisissez une méthode MFA pour vous authentifier, puis choisissez Suivant.

   > **Note**
   >
   > Vous pouvez voir un texte de remplacement, tel que Connectez-vous à l'aide de MFA, Dépanner votre dispositif d'authentification, ou Dépanner le MFA, mais les fonctionnalités sont les mêmes. Si vous ne pouvez pas utiliser d'autres facteurs d'authentification pour vérifier l'adresse e-mail et le numéro de téléphone du contact principal de votre compte, contactez AWS Support pour désactiver votre dispositif MFA.

3. Selon le type de MFA que vous utilisez, vous verrez une page différente, mais l'option Dépanner MFA fonctionne de la même manière. Sur la page Vérification supplémentaire requise ou sur la page Multi-factor d'authentification, choisissez Résoudre les problèmes liés à l'authentification multifacteur.
4. Si besoin, saisissez à nouveau votre mot de passe et choisissez Sign in (Connexion).
5. Sur la page Dépanner votre dispositif d'authentification, dans la section Connectez-vous à l'aide d'autres facteurs d'authentification, choisissez Connectez-vous à l'aide d'autres facteurs.
6. Sur la page Connectez-vous à l'aide d'autres facteurs d'authentification, authentifiez votre compte en vérifiant l'adresse e-mail, puis choisissez Envoyer un e-mail de vérification.
7. Vérifiez si l'e-mail qui vous Compte AWS est associé n'a pas reçu de message d'Amazon Web Services (recover-mfa-no-reply@verify.signin.aws). Suivez les instructions de l'e-mail.

   Si vous ne voyez pas d'e-mail dans votre boîte de réception, vérifiez le dossier des courriers indésirables, ou revenez à votre navigateur et choisissez Resend the email (Renvoyer l'e-mail).

8. Une fois que vous avez confirmé votre adresse e-mail, vous pouvez continuer à authentifier votre compte. Pour confirmer votre numéro de téléphone de contact principal, choisissez Call me now (Appelez-moi maintenant).
9. Répondez à l'appel AWS et, lorsque vous y êtes invité, entrez le numéro à 6 chiffres du AWS site Web sur le clavier de votre téléphone.

   Si vous ne recevez aucun appel de AWS, choisissez Se connecter pour vous reconnecter à la console et recommencer. Ou consultez la section Appareil Multi-Factor d'authentification perdu ou inutilisable (MFA) pour contacter le support technique pour obtenir de l'aide.
10. Une fois que vous avez confirmé votre numéro de téléphone, vous pouvez vous connecter à votre compte en choisissant Sign in to the console (Se connecter à la console).
11. L'étape suivante varie selon le type de MFA que vous utilisez :

    - Si vous utilisez un dispositif MFA virtuel, supprimez le compte de votre périphérique. Ensuite, accédez à la page Informations d'identification de sécuritéAWS et supprimez l'ancienne entité de dispositif virtuel MFA avant d'en créer une nouvelle.
    - Pour une clé de sécurité FIDO, consultez la page Informations d'identification AWS de sécurité et désactivez l'ancienne clé de sécurité FIDO avant d'en activer une nouvelle.
    - Si vous utilisez un jeton TOTP matériel, contactez le fournisseur tiers pour qu'il dépanne ou remplace le dispositif. Vous pouvez continuer à vous connecter à l'aide d'autres facteurs d'authentification jusqu'à ce que vous receviez votre nouveau périphérique. Une fois que vous avez le nouveau périphérique MFA matériel, consultez la page Informations d'identification AWS de sécurité et supprimez l'ancien périphérique MFA.

> **Note**
>
> Vous n'êtes pas obligé de remplacer un dispositif MFA perdu ou volé par le même type de périphérique. Par exemple, si vous cassez votre clé de sécurité FIDO et en commandez une nouvelle, vous pouvez utiliser un dispositif MFA virtuel ou un jeton TOTP matériel jusqu'à réception de la nouvelle clé FIDO.

> **Important**
>
> Si votre dispositif MFA est manquant ou volé, modifiez votre mot de passe d'utilisateur racine après vous être connecté et avoir mis en place votre dispositif MFA de remplacement. Un

attaquant pourrait avoir volé le dispositif d'authentification et pourrait également détenir votre mot de passe actuel. Pour de plus amples informations, veuillez consulter Changez le mot de passe pour Utilisateur racine d'un compte AWS.

Récupération d'un dispositif MFA d'utilisateur IAM

Si vous êtes un utilisateur IAM et que vous ne pouvez pas vous connecter par MFA, vous ne pouvez pas récupérer un dispositif MFA par vous-même. Vous devez contacter un administrateur pour désactiver le périphérique. Ensuite, vous pouvez activer un nouveau périphérique.

Pour obtenir de l'aide pour un dispositif MFA en tant qu'utilisateur IAM

1. Contactez l' AWS administrateur ou toute autre personne qui vous a fourni le nom d'utilisateur et le mot de passe de l'utilisateur IAM. L'administrateur doit désactiver le dispositif MFA, comme décrit dans Désactivation d'un dispositif MFA, afin que vous puissiez vous connecter.
2. L'étape suivante varie selon le type de MFA que vous utilisez :

   - Si vous utilisez un dispositif MFA virtuel, supprimez le compte de votre périphérique. Activez ensuite le périphérique virtuel comme décrit dans Attribuez un périphérique MFA virtuel dans le Console de gestion AWS.
   - Si vous utilisez une clé de sécurité FIDO, contactez le fournisseur tiers pour qu'il remplace le dispositif. Lorsque vous recevez la nouvelle clé de sécurité FIDO, activez-la comme décrit dans Attribuez un mot de passe ou une clé de sécurité dans le Console de gestion AWS.
   - Si vous utilisez un jeton TOTP matériel, contactez le fournisseur tiers pour qu'il dépanne ou remplace le dispositif. Une fois que vous avez le nouveau dispositif MFA, activez-le comme décrit dans Attribuez un jeton TOTP matériel dans Console de gestion AWS.

> **Note**
>
> Vous n'êtes pas obligé de remplacer un dispositif MFA perdu ou volé par le même type de périphérique. Vous pouvez avoir jusqu'à huit dispositifs MFA, quelle que soit leur combinaison. Par exemple, si vous cassez votre clé de sécurité FIDO et en commandez une nouvelle, vous pouvez utiliser un dispositif MFA virtuel ou un jeton TOTP matériel jusqu'à réception de la nouvelle clé FIDO.

3. Si votre dispositif MFA est perdu ou volé, modifiez votre mot de passe , au cas où un pirate informatique aurait volé le dispositif d'authentification et détiendrait également votre mot de passe actuel. Pour de plus amples informations, veuillez consulter Gérer les mots de passe des utilisateurs IAM.

Accès sécurisé aux API avec MFA

Avec les politiques IAM, vous pouvez spécifier quelles opérations d'API un utilisateur est autorisé à appeler. Vous pouvez renforcer la sécurité en exigeant des utilisateurs qu'ils s'authentifient à l'aide d'une authentification multifactorielle (MFA) avant de les autoriser à effectuer des actions particulièrement sensibles.

Par exemple, vous disposez peut-être d'une politique qui autorise l'utilisateur à exécuter les actions Amazon EC2 `RunInstances`, `DescribeInstances` et `StopInstances`. Mais vous souhaiterez peut-être restreindre une action destructrice, par exemple, `TerminateInstances` et vous assurer que les utilisateurs ne peuvent effectuer cette action que s'ils s'authentifient auprès d'un appareil AWS MFA.

Rubriques

- Présentation de
- Scénario : protection MFA pour la délégation entre comptes
- Scénario : protection MFA pour l'accès aux opérations d'API du compte actuel
- Scénario : protection MFA des ressources disposant de politiques basées sur les ressources

Présentation de

L'ajout d'une protection MFA aux opérations d'API nécessite les tâches suivantes :

1. L'administrateur configure un périphérique AWS MFA pour chaque utilisateur qui doit effectuer des demandes d'API nécessitant une authentification MFA. Pour de plus amples informations, veuillez consulter AWS Multi-factor authentification dans IAM.
2. L'administrateur crée des politiques pour les utilisateurs qui incluent un `Condition` élément qui vérifie si l'utilisateur s'est authentifié auprès d'un périphérique AWS MFA.
3. L'utilisateur appelle l'une des opérations d' AWS STS API qui prennent en charge les paramètres MFA : AssumeRole ou GetSessionToken. Dans le cadre de l'appel, l'utilisateur inclut l'identifiant du périphérique associé à l'utilisateur. L'utilisateur inclut également le mot de passe TOTP (Time-

based One-Time Password) que le périphérique génère. Dans tous les cas, l'utilisateur récupère les informations d'identification de sécurité temporaires qu'il peut ensuite utiliser pour effectuer des demandes supplémentaires à AWS.

> **Note**
>
> La protection MFA des opérations d'API d'un service est uniquement disponible si le service prend en charge les informations d'identification de sécurité temporaires. Pour connaître la liste de ces services, veuillez consulter Utilisation d'informations d'identification de sécurité temporaires pour accéder à AWS.

Si l'autorisation échoue, AWS renvoie un message d'erreur d'accès refusé (comme c'est le cas pour tout accès non autorisé). Lorsque les politiques d' MFA-protected API sont en place, AWS refuse l'accès aux opérations d'API spécifiées dans les politiques si l'utilisateur tente d'appeler une opération d'API sans authentification MFA valide. L'opération est également refusée si l'horodatage de la demande pour l'opération d'API se situe en dehors de la plage autorisée spécifiée dans la politique. L'utilisateur doit être à nouveau authentifié avec l'authentification MFA en demandant de nouvelles informations d'identification de sécurité temporaires avec un code MFA et un numéro de série de périphérique.

Politiques IAM avec des conditions MFA

Les politiques avec des conditions MFA peuvent être attachées à ce qui suit :

- Un utilisateur ou un groupe IAM
- Une ressource telle qu'un compartiment Amazon S3, une file d'attente Amazon SQS ou une rubrique Amazon SNS
- La politique d'approbation d'un rôle IAM qui peut être endossée par un utilisateur

Vous pouvez utiliser une condition MFA d'une politique pour vérifier les propriétés suivantes :

- Existence : pour vérifier simplement que l'utilisateur s'est authentifié par MFA, vérifiez que la clé `aws:MultiFactorAuthPresent` est `True` dans une condition `Bool`. La clé n'est présente que lorsque l'utilisateur s'authentifie avec des informations d'identification à court terme. Long-term les informations d'identification, telles que les clés d'accès, n'incluent pas cette clé.

- Durée : si vous voulez octroyer l'accès uniquement dans un délai spécifié après l'authentification MFA, utilisez un type de condition numérique pour comparer l'âge de la clé `aws:MultiFactorAuthAge` à une valeur (par exemple 3 600 secondes). Notez que la clé `aws:MultiFactorAuthAge` n'est pas présente si l'authentification MFA n'a pas été utilisée.

L'exemple suivant présente la politique d'approbation d'un rôle IAM incluant une condition MFA pour tester l'existence de l'authentification MFA. Grâce à cette politique, les utilisateurs de la zone Compte AWS spécifiée dans l'`Principal`élément (à `ACCOUNT-B-ID` remplacer par un Compte AWS identifiant valide) peuvent assumer le rôle auquel cette stratégie est attachée. Toutefois, ces utilisateurs peuvent endosser le rôle uniquement s'ils se sont authentifiés à l'aide de l'authentification MFA.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "ACCOUNT-B-ID"},
    "Action": "sts:AssumeRole",
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }
}
```

Pour plus d'informations sur les types de conditions pour l'authentification MFA, consultez AWS clés de contexte de condition globale, Opérateurs de condition numériques et Opérateur de condition pour vérifier l'existence de clés de condition

Choisir entre GetSessionToken et AssumeRole

AWS STS fournit deux opérations d'API qui permettent aux utilisateurs de transmettre des informations MFA : `GetSessionToken` et`AssumeRole`. L'opération d'API que l'utilisateur appelle pour obtenir des informations d'identification de sécurité temporaires dépend du scénario applicable parmi les suivants :

Utilisez **`GetSessionToken`** pour les scénarios suivants :

- Appelez les opérations d'API qui accèdent aux ressources de la même manière Compte AWS que l'utilisateur IAM qui fait la demande. Notez que les informations d'identification temporaires issues d'une `GetSessionToken` demande peuvent accéder aux opérations IAM et AWS STS API uniquement si vous incluez des informations MFA dans la demande d'informations d'identification. Du fait que les informations d'identification temporaires renvoyées par `GetSessionToken` incluent des informations d'authentification MFA, vous pouvez vérifier l'authentification MFA dans les opérations d'API individuelles effectuées par les informations d'identification.
- Accédez aux ressources protégées par des politiques basées sur les ressources qui incluent une condition MFA.

Le but principal de l'opération `GetSessionToken` est d'authentifier l'utilisateur à l'aide de l'authentification MFA. Vous ne pouvez pas utiliser de stratégies pour contrôler les opérations d'authentification.

Utilisez **`AssumeRole`** pour les scénarios suivants :

- Appeler des opérations d'API qui accèdent à des ressources dans le même Compte AWS ou dans un compte différent. Les appels d'API peuvent inclure n'importe quel IAM ou AWS STS API. Notez que pour protéger l'accès vous devez appliquer l'authentification MFA au moment où l'utilisateur endosse le rôle. Les informations d'identification temporaires renvoyées par `AssumeRole` n'incluent pas d'informations d'authentification MFA dans ce contexte ; vous ne pouvez donc pas vérifier les opérations d'API individuelles pour l'authentification MFA. C'est pourquoi vous devez utiliser `GetSessionToken` pour restreindre l'accès aux ressources protégées par des politiques basées sur les ressources.

> **Note**
>
> AWS CloudTrail les journaux contiendront des informations MFA lorsque l'utilisateur IAM se connecte avec MFA. Si l'utilisateur IAM assume un rôle IAM, il CloudTrail enregistre également les `sessionContext` attributs des actions effectuées `mfaAuthenticated: true` à l'aide du rôle assumé. Cependant, la CloudTrail journalisation est distincte de ce qu'IAM exige lorsque les appels d'API sont effectués avec les informations d'identification du rôle assumé. Pour en savoir plus, consultez Élément CloudTrail userIdentity.

Vous trouverez des détails sur l'implémentation de ces scénarios ultérieurement dans ce document.

Points importants concernant MFA-protected l'accès aux API

Il est important de comprendre les aspects suivants relatifs à la protection MFA pour les opérations d'API :

- La protection MFA est uniquement disponible avec les informations d'identification de sécurité temporaires que vous devez obtenir avec `AssumeRole` ou `GetSessionToken`.
- Vous ne pouvez pas utiliser MFA-protected l'accès API avec des Utilisateur racine d'un compte AWS informations d'identification.
- Vous ne pouvez pas utiliser MFA-protected l'accès API avec des clés de sécurité U2F.
- Les utilisateurs fédérés ne peuvent pas se voir attribuer un appareil MFA à utiliser avec les AWS services. Ils ne peuvent donc pas accéder aux AWS ressources contrôlées par MFA. (Voir le point suivant.)
- Les autres opérations d' AWS STS API qui renvoient des informations d'identification temporaires ne prennent pas en charge le MFA. Pour `AssumeRoleWithWebIdentity` et`AssumeRoleWithSAML`, l'utilisateur est authentifié par un fournisseur externe et AWS ne peut pas déterminer si ce fournisseur a requis le MFA. Pour `GetFederationToken`, l'authentification MFA n'est pas nécessairement associée à un utilisateur spécifique.
- De même, les informations d'identification à long terme (clés d'accès utilisateur IAM et clés d'accès utilisateur root) ne peuvent pas être utilisées avec MFA-protected l'accès à l'API car elles n'expirent pas.
- `AssumeRole` et `GetSessionToken` peuvent également être appelées sans informations MFA. Dans ce cas, le principal récupère les informations d'identification de sécurité temporaires, mais les informations de session de ces informations d'identification temporaires n'indiquent pas que l'utilisateur s'est authentifié avec l'authentification MFA.
- Pour établir la protection MFA pour les opérations d'API, ajoutez des conditions d'authentification MFA aux politiques. Une politique doit inclure la clé de condition `aws:MultiFactorAuthPresent` pour imposer l'utilisation de l'authentification MFA. Pour la délégation entre comptes, la politique d'approbation du rôle doit inclure la clé de condition.
- Lorsque vous autorisez une autre personne Compte AWS à accéder aux ressources de votre compte, la sécurité de vos ressources dépend de la configuration du compte sécurisé (l'autre compte, pas le vôtre). Cela est vrai même lorsque vous imposez une authentification multi-facteur. Toutes les identités du compte approuvé ayant l'autorisation de créer des dispositifs MFA virtuels peut créer une demande de MFA pour satisfaire cette partie de la politique d'approbation de votre rôle. Avant d'autoriser les membres d'un autre compte à accéder à vos AWS ressources qui nécessitent une authentification multifactorielle, vous devez vous assurer que le propriétaire du

compte de confiance suit les meilleures pratiques de sécurité. Par exemple, le compte approuvé doit restreindre l'accès aux opérations d'API sensibles, telles que les opérations d'API de gestion de dispositif MFA, à des identités approuvées spécifiques.

- Si une politique inclut une condition MFA, une demande est refusée si les utilisateurs n'ont pas été authentifiés par MFA ou s'ils fournissent un identifiant de dispositif MFA ou un TOTP non valide.

Scénario : protection MFA pour la délégation entre comptes

Dans ce scénario, vous souhaitez déléguer l'accès aux utilisateurs IAM d'un autre compte, mais uniquement si les utilisateurs sont authentifiés auprès d'un périphérique AWS MFA. Pour plus d'informations sur la délégation entre comptes, consultez Termes et concepts relatifs aux rôles.

Imaginons que vous disposiez d'un compte A (le compte d'approbation qui est titulaire de la ressource auxquelles les utilisateurs ont accès), avec l'utilisateur IAM Anaya qui dispose de l'autorisation administrateur. Elle souhaite accorder l'accès à l'utilisateur Richard au compte B (le compte approuvé), mais veut s'assurer que Richard s'est authentifié avec MFA avant d'endosser le rôle.

1. Dans le compte de confiance A, Anaya crée un rôle IAM nommé `CrossAccountRole` et définit le principal de la politique de confiance du rôle sur l'ID de compte du compte B. La politique de confiance autorise l'action. AWS STS `AssumeRole` Anaya ajoute également une condition MFA à la politique d'approbation, comme dans l'exemple suivant.

   JSON

   ```
   {
     "Version":"2012-10-17",
     "Statement": {
       "Effect": "Allow",
       "Principal": {"AWS": "ACCOUNT-B-ID"},
       "Action": "sts:AssumeRole",
       "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
     }
   }
   ```

2. Anaya ajoute une politique d'autorisations au rôle qui spécifie ce que le rôle est autorisé à faire. La politique d'autorisations d'un rôle avec protection MFA est identique à toutes les politiques d'autorisation de rôle. L'exemple suivant montre la politique qu'Anaya ajoute au rôle : elle permet à un utilisateur endossant le rôle d'exécuter toutes les actions Amazon DynamoDB sur la table

`Books` dans le compte A. Cette politique permet également l'action `dynamodb:ListTables`, qui est nécessaire pour effectuer des actions dans la console.

> **Note**
>
> La politique d'autorisations n'inclut pas de condition MFA. Il est important de comprendre que l'authentification MFA sert uniquement à déterminer si un utilisateur peut endosser le rôle. Une fois que l'utilisateur endosse le rôle, aucune autre vérification MFA n'est effectuée.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TableActions",
            "Effect": "Allow",
            "Action": "dynamodb:*",
            "Resource": "arn:aws:dynamodb:*:111122223333:table/Books"
        },
        {
            "Sid": "ListTables",
            "Effect": "Allow",
            "Action": "dynamodb:ListTables",
            "Resource": "*"
        }
    ]
}
```

3. Dans le compte sécurisé B, l'administrateur s'assure que l'utilisateur IAM Richard est configuré avec un périphérique AWS MFA et qu'il connaît l'ID de ce périphérique. L'ID du périphérique est le numéro de série s'il s'agit d'un dispositif MFA matériel, ou l'ARN du périphérique s'il s'agit d'un dispositif MFA virtuel.
4. Dans le compte B, l'administrateur attache la politique suivante à l'utilisateur Richard (ou à un groupe dont il est membre) qui l'autorise à appeler l'action `AssumeRole`. La ressource est définie sur l'ARN du rôle qu'Anaya a créé à l'étape 1. Notez que cette politique n'inclut aucune condition MFA.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole"
            ],
            "Resource": [
                "arn:aws:iam::111122223333:role/CrossAccountRole"
            ]
        }
    ]
}
```

5. Dans le compte B, Richard (ou une application qu'il exécute) appelle `AssumeRole`. L'appel d'API inclut l'ARN du rôle à endosser (`arn:aws:iam::`*`ACCOUNT-A-ID`*`:role/CrossAccountRole`), l'ID du dispositif MFA et le TOTP actuel que Richard obtient de ce périphérique.

   Lorsque Richard appelle`AssumeRole`, il AWS détermine s'il possède des informations d'identification valides, y compris l'exigence du MFA. Le cas échéant, Richard réussit à endosser le rôle et peut exécuter n'importe quelle action DynamoDB sur la table nommée `Books` dans le compte A tout en utilisant les informations d'identification temporaires du rôle.

   Pour obtenir un exemple d'un programme qui appelle `AssumeRole`, consultez Appels AssumeRole avec authentification MFA.

Scénario : protection MFA pour l'accès aux opérations d'API du compte actuel

Dans ce scénario, vous devez vous assurer qu'un utilisateur de votre réseau ne Compte AWS peut accéder à des opérations d'API sensibles que s'il est authentifié à l'aide d'un appareil AWS MFA.

Imaginons que vous ayez un compte A contenant un groupe de développeurs ayant besoin d'utiliser des instances EC2. Les développeurs standard peuvent utiliser les instances, mais ils n'ont pas l'autorisation d'utiliser les actions `ec2:StopInstances` ou `ec2:TerminateInstances`. Vous souhaitez limiter ces actions « destructives » réalisées avec des actions à quelques utilisateurs approuvés uniquement, vous ajoutez donc la protection MFA à la politique qui autorise ces actions Amazon EC2 sensibles.

Dans ce scénario, l'un des utilisateurs approuvé s'appelle Sofía. L'utilisatrice Anaya est administratrice du compte A.

1. Anaya s'assure que Sofía est configurée avec un appareil AWS MFA et que Sofía connaît l'identifiant de l'appareil. L'ID du périphérique est le numéro de série s'il s'agit d'un dispositif MFA matériel, ou l'ARN du périphérique s'il s'agit d'un dispositif MFA virtuel.
2. Anaya crée un groupe appelé `EC2-Admins` et ajoute l'utilisatrice Sofía au groupe.
3. Anaya attache la politique suivante au groupe `EC2-Admins`. Cette politique autorise également les utilisateurs à appeler les actions `StopInstances` et `TerminateInstances` Amazon EC2 uniquement si l'utilisateur s'est authentifié à l'aide de l'authentification MFA.

   JSON

   ```
   {
     "Version":"2012-10-17",
     "Statement": [{
       "Effect": "Allow",
       "Action": [
         "ec2:StopInstances",
         "ec2:TerminateInstances"
       ],
       "Resource": ["*"],
       "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
     }]
   }
   ```

4. > **Note**
   >
   > Pour que cette politique prenne effet, les utilisateurs doivent d'abord se déconnecter et se connecter à nouveau.

   Si l'utilisatrice Sofía a besoin d'arrêter ou de résilier une instance Amazon EC2, elle (ou une application qu'elle exécute) appelle `GetSessionToken`. Cette opération d'API transmet l'ID du dispositif MFA et le TOTP actuel que Sofía obtient de son périphérique.
5. L'utilisatrice Sofía (ou une application qu'elle utilise) utilise les informations d'identification temporaires fournies par `GetSessionToken` pour appeler l'action `StopInstances` ou `TerminateInstances` Amazon EC2.

Pour obtenir un exemple d'un programme qui appelle `GetSessionToken`, consultez Appels GetSessionToken avec authentification MFA ci-après dans ce document.

Scénario : protection MFA des ressources disposant de politiques basées sur les ressources

Dans ce scénario, vous êtes le propriétaire d'un compartiment S3, d'une file d'attente SQS ou d'une rubrique SNS. Vous devez vous assurer que tous les utilisateurs Compte AWS qui accèdent à la ressource sont authentifiés par un périphérique AWS MFA.

Ce scénario illustre un processus permettant de fournir une protection MFA entre comptes sans nécessiter que les utilisateurs endossent d'abord un rôle. Dans ce cas, l'utilisateur peut accéder à la ressource s'il répond à trois conditions : l'utilisateur doit être authentifié par l'authentification MFA, être en mesure d'obtenir des informations d'identification temporaires de `GetSessionToken` et disposer d'un compte approuvé par la politique de la ressource.

Imaginons que vous vous trouvez dans le compte A et que vous créez un compartiment S3. Vous souhaitez accorder l'accès à ce compartiment à des utilisateurs appartenant à plusieurs entités différentes Comptes AWS, mais uniquement s'ils sont authentifiés par MFA.

Dans ce scénario, l'utilisatrice Anaya est une administratrice du compte A. L'utilisateur Nikhil est un utilisateur IAM du compte C.

1. Dans le compte A, Anaya crée un compartiment appelé `Account-A-bucket`.
2. Anaya ajoute la politique de compartiment au compartiment. La politique autorise n'importe quel utilisateur du compte A, du compte B ou du compte C à exécuter les actions `PutObject` et `DeleteObject` Amazon S3 dans le compartiment. La politique inclut une condition MFA.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"AWS": [
      "ACCOUNT-A-ID",
      "ACCOUNT-B-ID",
      "ACCOUNT-C-ID"
    ]},
    "Action": [
      "s3:PutObject",
```

```
      "s3:DeleteObject"
    ],
    "Resource": ["arn:aws:s3:::ACCOUNT-A-BUCKET-NAME/*"],
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }]
}
```

> **Note**
>
> Amazon S3 fournit une fonction de suppression MFA (MFA Delete) pour l'accès au compte racine (uniquement). Vous pouvez activer la fonction de suppression MFA d'Amazon S3 lorsque vous définissez l'état de la gestion des versions du compartiment. Amazon S3 MFA Delete ne peut pas être appliqué à un utilisateur IAM et est géré indépendamment de l'accès à MFA-protected l'API. Un utilisateur IAM disposant des autorisations nécessaires pour supprimer un compartiment ne peut pas le faire si la fonction de suppression MFA Amazon S3 est activée. Pour plus d'informations sur la fonction Supprimer MFA d'Amazon S3, consultez la section Supprimer MFA.

3. Dans le compte C, un administrateur vérifie que l'utilisateur Nikhil est configuré avec un dispositif MFA AWS et qu'il connaît l'ID du dispositif. L'ID du périphérique est le numéro de série s'il s'agit d'un dispositif MFA matériel, ou l'ARN du périphérique s'il s'agit d'un dispositif MFA virtuel.
4. Dans le compte C, Nikhil (ou une application qu'il exécute) appelle `GetSessionToken`. L'appel inclut l'ID ou l'ARN du dispositif MFA et le TOTP actuel que Nikhil obtient de ce périphérique.
5. Nikhil (ou une application qu'il utilise) utilise les informations d'identification temporaires renvoyées par `GetSessionToken` pour appeler l'action `PutObject` Amazon S3 afin de télécharger un fichier dans `Account-A-bucket`.

   Pour obtenir un exemple d'un programme qui appelle `GetSessionToken`, consultez Appels GetSessionToken avec authentification MFA ci-après dans ce document.

> **Note**
>
> Les informations d'identification temporaires renvoyées par `AssumeRole` ne fonctionnent pas dans ce cas. Bien que l'utilisateur puisse fournir les informations MFA lui permettant d'endosser un rôle, les informations d'identification temporaires renvoyées par `AssumeRole` n'incluent par les informations MFA. Ces informations sont requises pour satisfaire la condition MFA de la politique.

Exemple de code : demande d'informations d'identification avec l'authentification multifacteur

Les exemples suivants montrent comment appeler les opérations `GetSessionToken` et `AssumeRole` et transmettre les paramètres d'authentification MFA. Aucune autorisation n'est requise pour appeler `GetSessionToken`, toutefois, vous devez disposer d'une politique vous permettant d'appeler `AssumeRole`. Les informations d'identification renvoyées sont ensuite utilisées afin de répertorier tous les compartiments S3 dans le compte.

Appels GetSessionToken avec authentification MFA

L'exemple suivant explique comment appeler `GetSessionToken` et transmettre les informations d'authentification MFA. Les informations d'identification de sécurité temporaires renvoyées par l'opération `GetSessionToken` sont ensuite utilisées pour répertorier tous les compartiments S3 dans le compte.

La politique associée à l'utilisateur exécutant ce code (ou à un groupe dont fait partie l'utilisateur) fournit les autorisations pour les informations d'identification temporaires renvoyées. Dans cet exemple de code, la politique doit autoriser l'utilisateur à demander l'opération `ListBuckets` Amazon S3.

Les exemples de code suivants illustrent comment utiliser `GetSessionToken`.

CLI

AWS CLI

Pour obtenir un ensemble d'informations d'identification à court terme d'une identité IAM

La commande `get-session-token` suivante extrait un ensemble d'informations d'identification à court terme pour l'identité IAM qui effectue l'appel. Les informations d'identification obtenues peuvent être utilisées pour les requêtes où l'authentification multifactorielle (MFA) est requise par la politique. Les informations d'identification expirent 15 minutes après leur création.

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

Sortie :

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

Pour plus d'informations, consultez Demande d'informations d'identification temporaires de sécurité dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir GetSessionTokenla section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : renvoie une instance **`Amazon.RuntimeAWSCredentials`** contenant des informations d'identification temporaires valables pendant une période définie. Les informations d'identification utilisées pour demander des informations d'identification temporaires sont déduites des paramètres par défaut actuels du shell. Pour spécifier d'autres informations d'identification, utilisez les SecretKey paramètres - ProfileName ou - AccessKey /-.

```
Get-STSSessionToken
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemple 2 : renvoie une instance **Amazon.RuntimeAWSCredentials** contenant des informations d'identification temporaires valides pour une heure. Les informations d'identification utilisées pour effectuer la demande sont obtenues à partir du profil spécifié.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemple 3 : renvoie une instance **Amazon.RuntimeAWSCredentials** contenant des informations d'identification temporaires valables pendant une heure en utilisant le numéro d'identification du dispositif MFA associé au compte dont les informations d'identification sont spécifiées dans le profil « myprofilename » et la valeur fournie par le dispositif.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- Pour plus de détails sur l'API, reportez-vous GetSessionTokenà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : renvoie une instance **Amazon.RuntimeAWSCredentials** contenant des informations d'identification temporaires valables pendant une période définie. Les informations d'identification utilisées pour demander des informations d'identification

temporaires sont déduites des paramètres par défaut actuels du shell. Pour spécifier d'autres informations d'identification, utilisez les SecretKey paramètres - ProfileName ou - AccessKey /-.

```
Get-STSSessionToken
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemple 2 : renvoie une instance **Amazon.RuntimeAWSCredentials** contenant des informations d'identification temporaires valides pour une heure. Les informations d'identification utilisées pour effectuer la demande sont obtenues à partir du profil spécifié.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemple 3 : renvoie une instance **Amazon.RuntimeAWSCredentials** contenant des informations d'identification temporaires valables pendant une heure en utilisant le numéro d'identification du dispositif MFA associé au compte dont les informations d'identification sont spécifiées dans le profil « myprofilename » et la valeur fournie par le dispositif.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- Pour plus de détails sur l'API, reportez-vous GetSessionTokenà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Obtenez un jeton de session en transmettant un jeton MFA et utilisez-le pour répertorier les compartiments Amazon S3 pour le compte.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
```

```
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Pour plus de détails sur l'API, consultez GetSessionTokenle AWS manuel de référence de l'API SDK for Python (Boto3).

Appels AssumeRole avec authentification MFA

Les exemples suivants, expliquent comment appeler `AssumeRole` et transmettre les informations d'authentification MFA. Les informations d'identification de sécurité temporaires renvoyées par `AssumeRole` sont ensuite utilisées pour répertorier tous les compartiments Amazon S3 du compte.

Pour plus d'informations sur ce scénario, consultez Scénario : protection MFA pour la délégation entre comptes.

Les exemples de code suivants illustrent comment utiliser `AssumeRole`.

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;

namespace AssumeRoleExample
{
    class AssumeRole
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
        ///
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        // A region property may be used if the profile or credentials loaded do
 not specify a region,
        // or to use a specific region.
        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
```

```
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"Original Caller: {caller.Arn}");

            // Create the request to use with the AssumeRoleAsync call.
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);

            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials, REGION);

            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
    }
}
```

- Pour plus de détails sur l'API, voir AssumeRolela section Référence des AWS SDK pour .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
##############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
```

```
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, voir AssumeRolela section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
                             const Aws::String &externalId,
                             Aws::Auth::AWSCredentials &credentials,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;

    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
```

```
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, voir AssumeRolela section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour endosser un rôle

La commande `assume-role` suivante extrait un ensemble d'informations d'identification à court terme pour le rôle IAM `s3-access-example`.

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

Sortie :

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
    },
    "Credentials": {
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
```

```
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

La sortie de la commande contient une clé d'accès, une clé secrète et un jeton de session que vous pouvez utiliser pour vous authentifier auprès d' AWS.

Pour l'utilisation de la AWS CLI, vous pouvez configurer un profil nommé associé à un rôle. Lorsque vous utilisez le profil, la AWS CLI appelle assume-role et gère les informations d'identification pour vous. Pour plus d'informations, consultez la section Utiliser un rôle IAM dans la AWS CLI du Guide de l'utilisateur de la AWS CLI.

- Pour plus de détails sur l'API, voir AssumeRolela section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
import java.time.Instant;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
```

```
 * example:
 *
 * {
 * "Version":"2012-10-17",
 * "Statement": [
 * {
 * "Effect": "Allow",
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code example>"
 * },
 * "Action": "sts:AssumeRole"
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
 * Role" in the AWS Directory Service guide.
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s

                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
 assume (for example, arn:aws:iam::000008047983:role/s3role).\s
                    roleSessionName - An identifier for the assumed role session
 (for example, mysession).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }
```

```
        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
                .region(region)
                .build();

        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }

    public static void assumeGivenRole(StsClient stsClient, String roleArn,
 String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
 DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
 exTime);

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, voir AssumeRolela section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez le client.

```
import { STSClient } from "@aws-sdk/client-sts";
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

Assumez le rôle IAM.

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
```

```
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Pour plus de détails sur l'API, voir AssumeRolela section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });

//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
```

```
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
});

//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);

  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- Pour plus de détails sur l'API, voir AssumeRolela section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : renvoie un ensemble d'informations d'identification temporaires (clé d'accès, clé secrète et jeton de session) qui peuvent être utilisées pendant une heure pour accéder à AWS des ressources auxquelles l'utilisateur demandeur n'a pas normalement accès. Les informations d'identification renvoyées disposent des autorisations autorisées par la stratégie d'accès du rôle assumé et par la politique fournie (vous ne pouvez pas utiliser la politique fournie pour accorder des autorisations supérieures à celles définies par la stratégie d'accès du rôle assumé).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Exemple 2 : renvoie un ensemble d'informations d'identification temporaires, valides pendant une heure, dotées des mêmes autorisations que celles définies dans la stratégie d'accès du rôle assumé.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Exemple 3 : renvoie un ensemble d'informations d'identification temporaires fournissant le numéro de série et le jeton généré par une MFA associée aux informations d'identification de l'utilisateur utilisées pour exécuter l'applet de commande.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Exemple 4 : renvoie un ensemble d'informations d'identification temporaires qui ont endossé un rôle défini dans le compte d'un client. Pour chaque rôle que le tiers peut assumer, le compte client doit créer un rôle à l'aide d'un identifiant qui doit être transmis dans le ExternalId paramètre - chaque fois que le rôle est assumé.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : renvoie un ensemble d'informations d'identification temporaires (clé d'accès, clé secrète et jeton de session) qui peuvent être utilisées pendant une heure pour accéder à AWS des ressources auxquelles l'utilisateur demandeur n'a pas normalement accès. Les informations d'identification renvoyées disposent des autorisations autorisées par la stratégie d'accès du rôle assumé et par la politique fournie (vous ne pouvez pas utiliser la politique fournie pour accorder des autorisations supérieures à celles définies par la stratégie d'accès du rôle assumé).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Exemple 2 : renvoie un ensemble d'informations d'identification temporaires, valides pendant une heure, dotées des mêmes autorisations que celles définies dans la stratégie d'accès du rôle assumé.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Exemple 3 : renvoie un ensemble d'informations d'identification temporaires fournissant le numéro de série et le jeton généré par une MFA associée aux informations d'identification de l'utilisateur utilisées pour exécuter l'applet de commande.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Exemple 4 : renvoie un ensemble d'informations d'identification temporaires qui ont endossé un rôle défini dans le compte d'un client. Pour chaque rôle que le tiers peut assumer, le compte client doit créer un rôle à l'aide d'un identifiant qui doit être transmis dans le ExternalId paramètre - chaque fois que le rôle est assumé.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Assumez un rôle IAM qui nécessite un jeton MFA et utilisez des informations d'identification temporaires pour répertorier les compartiments Amazon S3 pour le compte.

```
def list_buckets_from_assumed_role_with_mfa(
```

```
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Pour plus de détails sur l'API, consultez AssumeRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: 'create-use-assume-role-scenario'
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
```

```
  end
```

- Pour plus de détails sur l'API, voir AssumeRolela section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
        .load()
        .await;
    let client = Client::new(&local_config);
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
        Err(e) => println!("{:?}", e),
    }
```

```
}
```

- Pour plus de détails sur l'API, voir AssumeRolela section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSSTS

    public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
        async throws -> STSClientTypes.Credentials
    {
        let input = AssumeRoleInput(
            roleArn: role.arn,
            roleSessionName: sessionName
        )
        do {
            let output = try await stsClient.assumeRole(input: input)

            guard let credentials = output.credentials else {
                throw ServiceHandlerError.authError
            }

            return credentials
        } catch {
            print("Error assuming role: ", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section AWS SDK pour la référence de l'API Swift.

Informations d’identification spécifiques au service pour les utilisateurs IAM

Les informations d'identification spécifiques à un service sont des mécanismes d'authentification spécialisés conçus pour des services spécifiques AWS . Ces informations d'identification fournissent une authentification simplifiée par rapport aux AWS informations d'identification standard et sont adaptées aux exigences d'authentification de chaque AWS service. Contrairement aux clés d'accès, qui peuvent être utilisées pour plusieurs AWS services, les informations d'identification spécifiques à un service sont conçues pour être utilisées uniquement avec le service pour lequel elles ont été créées. Cette approche ciblée améliore la sécurité en limitant la portée des informations d’identification.

Les informations d’identification spécifiques à un service se composent généralement d’un nom d’utilisateur et d’un mot de passe ou de clés d’API spécialisées dont le format est conforme aux exigences du service concerné. Lorsque vous créez des informations d’identification spécifiques à un service, celles-ci sont actives par défaut et peuvent être utilisées immédiatement. Vous pouvez avoir un maximum de deux ensembles d’informations d’identification spécifiques au service pour chaque service pris en charge par utilisateur IAM. Cette limite vous permet de conserver un ensemble actif tout en passant à un nouveau set en cas de besoin. AWS prend actuellement en charge les informations d'identification spécifiques au service pour les services suivants :

Quand utiliser les informations d'identification spécifiques au service

Les informations d'identification spécifiques au service sont destinées à être compatibles avec des bibliothèques SDKs, des outils ou des applications tiers qui ne sont pas compatibles nativement avec les AWS informations d'identification, AWS SDKs ou. AWS APIs Ces cas d'utilisation incluent la migration vers des AWS services à partir d'une infrastructure auto-hébergée ou de services hébergés par d'autres fournisseurs.

Lorsque vous partez de zéro, et dans la mesure du possible, nous vous recommandons d'utiliser des informations d'identification AWS temporaires, telles que celles fournies par un rôle IAM, pour vous authentifier auprès d'un AWS service à l'aide d'un AWS SDK ou d'une bibliothèque qui prend en charge AWS les informations d'identification temporaires.

Changement des informations d’identification spécifiques au service

En tant que bonne pratique en matière de sécurité, renouvelez régulièrement les informations d’identification spécifiques aux services. Pour changer les informations d’identification sans perturber vos applications :

1. Créez un deuxième ensemble d’informations d’identification spécifiques au service pour le même service et le même utilisateur IAM
2. Mettez à jour toutes les applications afin qu’elles utilisent les nouvelles informations d’identification et vérifiez qu’elles fonctionnent correctement
3. Modifiez le statut des informations d’identification d’origine en « Inactif »
4. Vérifiez que toutes les applications fonctionnent toujours correctement
5. Supprimez les informations d’identification spécifiques au service inactif lorsque vous êtes certain qu’elles ne sont plus nécessaires.

Surveillance des informations d’identification spécifiques au service

Vous pouvez l'utiliser AWS CloudTrail pour surveiller l'utilisation des informations d'identification spécifiques au service dans votre AWS compte. Pour consulter les CloudTrail événements liés à l'utilisation des informations d'identification spécifiques à un service, consultez les CloudTrail journaux des événements provenant du service où les informations d'identification sont utilisées. Pour de plus amples informations, veuillez consulter Journalisation IAM et AWS STS Appels d'API avec AWS CloudTrail.

Pour plus de sécurité, envisagez de configurer des CloudWatch alarmes pour vous avertir de modèles d'utilisation spécifiques des informations d'identification susceptibles d'indiquer un accès non autorisé ou d'autres problèmes de sécurité. Pour plus d'informations, consultez la section Surveillance des fichiers CloudTrail CloudWatch journaux avec Amazon Logs dans le guide de AWS CloudTrail l'utilisateur.

Les rubriques suivantes fournissent des informations sur les informations d’identification spécifiques au service.

Rubriques

- Clés d'API pour AWS services
- Utilisation d’IAM avec Amazon Keyspaces (pour Apache Cassandra)

Clés d'API pour AWS services

Certains AWS services prennent en charge les clés d'API pour authentifier les demandes programmatiques en plus des informations d'identification IAM standard telles que les informations d'identification de sécurité temporaires et les clés d'accès à long terme. AWS propose deux types de clés d'API :

- Long-term Clés d'API : les clés Long-term d'API sont associées à un utilisateur IAM et générées à l'aide d'informations d'identification spécifiques au service IAM. Ces informations d'identification sont conçues pour être utilisées avec un seul AWS service, ce qui renforce la sécurité en limitant la portée des informations d'identification. Vous pouvez définir une date d'expiration pour la clé d'API à long terme. Pour générer des clés d'API à long terme, vous pouvez utiliser la console IAM ou spécifique au service AWS CLI, l'API ou l'API. AWS
- Short-term Clés d'API : une clé d'API à court terme est une URL pré-signée qui utilise AWS la version 4 de Signature. Short-term Les clés API partagent les mêmes autorisations et expirent que les informations d'identification de l'identité qui génère la clé API et sont valides jusqu'à 12 heures ou jusqu'à la fin de votre session de console, la durée la plus courte étant retenue. Vous pouvez utiliser la Claude Platform sur AWS console Amazon Bedrock/, Python et les packages d'autres langages de programmation pour générer des clés d'API à court terme. Pour plus d'informations, consultez Générer des clés d'API Amazon Bedrock pour accéder facilement à l'API Amazon Bedrock dans le guide de l'utilisateur Amazon Bedrock et Authentification dans le guide de l'Claude Platform sur AWS utilisateur.

> **Note**
>
> Long-term Les clés d'API présentent un risque de sécurité plus élevé que les clés d'API à court terme. Nous vous recommandons d'utiliser des clés d'API à court terme ou des informations d'identification de sécurité temporaires si possible. Si vous utilisez des clés API à long terme, nous vous recommandons de mettre en place des pratiques régulières de rotation des clés.

Services prenant en charge les clés d'API

Le tableau suivant répertorie les AWS services qui prennent en charge les clés d'API et le type de clé d'API pris en charge par chaque service.

#	Service	Long-term Clés d'API	Short-term Clés d'API	Attachement automatique des politiques gérées	Service-specific documentation
1	Amazon Bedroc	Oui	Oui	AmazonBedrockLimitedAccess	Utiliser une clé d'API Amazon Bedrock
2	Claude Platform sur AWS	Oui	Oui	AnthropicInferenceAccess	Authentification
3	Amazon CloudWatch	Oui	N/A	CloudWatchAPIKeyAccess	Configuration de l'authentification par jeton au porteur pour Metrics
4	CloudWatch Journaux Amazon	Oui	N/A	CloudWatchLogsAPIKeyAccess	Configuration de l'authentification par jeton du porteur

Lorsque vous générez une clé API à long terme pour un service, la politique AWS gérée correspondante est automatiquement associée à l'utilisateur IAM, ce qui lui permet d'accéder aux opérations principales de ce service. Si vous avez besoin d'un accès supplémentaire, vous pouvez modifier les autorisations de l'utilisateur IAM. Pour plus d’informations sur la modification des autorisations, consultez the section called “Ajouter ou supprimer des autorisations d'identité”.

Pour en savoir plus sur les clés d'API pour des services spécifiques, consultez les liens de Service-specific documentation dans le tableau ci-dessus.

Conditions requises pour les clés d'API à long terme

Avant de pouvoir générer une clé d'API à long terme dans la console IAM, vous devez remplir les conditions préalables suivantes :

- Un utilisateur IAM à associer à la clé d'API à long terme. Pour plus d'informations sur la création d'un utilisateur IAM, consultez the section called "Créez un utilisateur".
- Vous devez disposer des autorisations de politique IAM suivantes pour gérer les informations d'identification spécifiques au service pour un utilisateur IAM. L'exemple de politique accorde l'autorisation de créer, répertorier, mettre à jour, supprimer et réinitialiser les informations d'identification spécifiques au service. Remplacez la *username* valeur de l'élément Resource par le nom de l'utilisateur IAM pour lequel vous allez générer des clés d'API à long terme :

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ManageBedrockServiceSpecificCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:UpdateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ResetServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/username"
        }
    ]
}
```

Génération d'une clé API à long terme (console)

Pour générer une clé d'API à long terme pour un service spécifique dans la console IAM

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation de la console IAM, sélectionnez Utilisateurs.

3. Choisissez l'utilisateur IAM pour lequel vous souhaitez générer une clé API à long terme.
4. Choisissez l'onglet Informations d'identification de sécurité.
5. Dans la section Clés d'API, choisissez Générer une clé d'API.
6. Dans la liste déroulante des AWS services, choisissez le service auprès duquel vous souhaitez que la clé API s'authentifie.
7. Pour Epiration de la clé d'API, effectuez l'une des actions suivantes :
   - Choisissez une durée d'expiration de la clé API de 1 , 5, 30, 90 ou 365 jours.
   - Choisissez Durée personnalisée pour spécifier une date d'expiration personnalisée pour la clé d'API.
   - Choisissez N'expire jamais (non recommandé).
8. Choisissez Générer une clé d'API.
9. Copiez ou téléchargez votre clé d'API. C'est le seul moment où vous pouvez voir la valeur de la clé API.

> **Important**
>
> Stockez votre clé d'API en toute sécurité. Une fois la boîte de dialogue fermée, vous ne pouvez plus récupérer la clé d'API. Si vous perdez ou oubliez votre clé API, vous ne pourrez pas la récupérer. Générez plutôt une nouvelle clé API et désactivez l'ancienne clé.

Génération d'une clé API à long terme (AWS CLI)

Pour générer une clé API à long terme à l'aide de AWS CLI, procédez comme suit :

1. Créez un utilisateur IAM qui sera utilisé avec le service à l'aide de la commande https://awscli.amazonaws.com/v2/documentation/api/latest/reference/iam/create-user.html create-user :

```
aws iam create-user \
    --user-name APIKeyUser_1
```

2. Attachez la politique AWS gérée à l'utilisateur IAM à l'aide de la commande attach-user-policy.

   Pour Amazon Bedrock :

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/AmazonBedrockLimitedAccess
```

Pour Claude Platform sur AWS :

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/AnthropicInferenceAccess
```

Pour Amazon CloudWatch :

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/CloudWatchAPIKeyAccess
```

Pour Amazon CloudWatch Logs :

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/CloudWatchLogsAPIKeyAccess
```

3. Générez la clé API à long terme à l'aide de la commande create-service-specific-credentials.

Pour Amazon Bedrock :

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name bedrock.amazonaws.com \
    --credential-age-days 30
```

Pour Claude Platform sur AWS :

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name aws-external-anthropic.amazonaws.com \
    --credential-age-days 30
```

Pour Amazon CloudWatch :

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name cloudwatch.amazonaws.com \
```

```
    --credential-age-days 30
```

Pour Amazon CloudWatch Logs :

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name logs.amazonaws.com \
    --credential-age-days 30
```

> **Note**
>
> Le paramètre `--credential-age-days` est facultatif. Vous pouvez spécifier une valeur comprise entre 1 et 36 600 jours. Si vous omettez ce paramètre, la clé API n'expire pas.

La réponse `ServiceApiKeyValue` renvoyée est votre clé API à long terme pour le service concerné. Stockez la valeur `ServiceApiKeyValue` en toute sécurité, car vous ne pourrez pas la récupérer ultérieurement.

Répertorier les clés API à long terme (AWS CLI)

Pour répertorier les métadonnées des clés d'API à long terme pour un utilisateur spécifique, utilisez la commande https://awscli.amazonaws.com/v2/documentation/api/latest/reference/iam/list-service-specific-credentials.html list-service-specific-credentials avec le paramètre : `--user-name`

```
aws iam list-service-specific-credentials \
    --service-name bedrock.amazonaws.com \
    --user-name APIKeyUser_1
```

> **Note**
>
> `bedrock.amazonaws.com`Remplacez-le par le nom de service approprié (par exemple, `logs.amazonaws.com` pour Amazon CloudWatch Logs ou `aws-external-anthropic.amazonaws.com` pour Claude Platform sur AWS).

Pour répertorier toutes les métadonnées des clés d'API à long terme du compte, utilisez la commande https://awscli.amazonaws.com/v2/documentation/api/latest/reference/iam/list-service-specific-credentials.html list-service-specific-credentials avec le paramètre : `--all-users`

```
aws iam list-service-specific-credentials \
    --service-name bedrock.amazonaws.com \
    --all-users
```

Mettre à jour l'état de la clé API à long terme (AWS CLI)

Pour mettre à jour l'état d'une clé API à long terme, utilisez la commande https://awscli.amazonaws.com/v2/documentation/api/latest/reference/iam/update-service-specific-credential.html update-service-specific-credentials :

```
aws iam update-service-specific-credential \
    --user-name "APIKeyUser_1" \
    --service-specific-credential-id "ACCA1234EXAMPLE1234" \
    --status Inactive|Active
```

Génération d'une clé API à long terme (AWS API)

Vous pouvez utiliser les opérations d'API IAM suivantes pour gérer les clés d'API à long terme pour tous les services pris en charge :

- CreateServiceSpecificCredential
- ListServiceSpecificCredentials
- UpdateServiceSpecificCredential
- DeleteServiceSpecificCredential
- ResetServiceSpecificCredential

Short-term Clés d'API (certains services)

Short-term Les clés d'API sont actuellement prises en charge par certains services.

Pour plus d'informations sur la génération et l'utilisation de clés d'API à court terme avec Amazon Bedrock, consultez Générer une clé d'API dans le guide de l'utilisateur d'Amazon Bedrock.

Pour plus d'informations sur la génération et l'utilisation de clés d'API à court terme pour Claude Platform sur AWS, consultez la section Authentification dans le guide de Claude Platform sur AWS l'utilisateur.

Service-specific information

- Pour plus d'informations sur l'utilisation des clés d'API avec Amazon Bedrock, consultez la section Utiliser une clé d'API Amazon Bedrock dans le guide de l'utilisateur d'Amazon Bedrock.
- Pour plus d'informations sur l'utilisation des clés API avec Claude Platform sur AWS, consultez la section Authentification dans le Guide de Claude Platform sur AWS l'utilisateur.
- Pour plus d'informations sur l'utilisation des clés API avec Amazon CloudWatch, consultez Configuration de l'authentification par jeton au porteur pour les métriques dans le guide de l' CloudWatchutilisateur Amazon.
- Pour plus d'informations sur l'utilisation des clés API avec Amazon CloudWatch Logs, consultez la section Configuration de l'authentification par jeton du porteur dans le guide de l'utilisateur Amazon CloudWatch Logs.

Utilisation d'IAM avec Amazon Keyspaces (pour Apache Cassandra)

Amazon Keyspaces (pour Apache Cassandra) est un service de base de données compatible avec Apache Cassandra, évolutif, hautement disponible et géré. Vous pouvez accéder à Amazon Keyspaces par le biais du ou par Console de gestion AWS programme. Pour accéder à Amazon Keyspaces de manière programmatique avec des informations d'identification spécifiques au service, vous pouvez utiliser `cqlsh` ou des pilotes Cassandra open source. Les informations d'identification spécifiques à un service comprennent un nom d'utilisateur et un mot de passe similaires à ceux que Cassandra utilise pour l'authentification et la gestion des accès. Vous pouvez avoir un maximum de deux ensembles d'informations d'identification spécifiques au service pour chaque service pris en charge par utilisateur.

Pour accéder à Amazon Keyspaces par programmation à l'aide de clés d' AWS accès, vous pouvez utiliser le AWS SDK, le AWS Command Line Interface (AWS CLI) ou les pilotes Cassandra open source avec le plugin SigV4. Pour en savoir plus, consultez la section Création et configuration des AWS informations d'identification pour Amazon Keyspaces (pour Apache Cassandra) dans le guide du développeur Amazon Keyspaces (pour Apache Cassandra).

> **Note**
>
> Si vous envisagez d'interagir avec Amazon Keyspaces uniquement via la console, vous n'avez pas besoin de générer des informations d'identification spécifiques au service. Pour de plus amples informations, veuillez consulter la rubrique Accès à Amazon Keyspaces à l'aide de la console dans le Guide du développeur Amazon Keyspaces (pour Apache Cassandra).

Pour de plus amples informations sur les autorisations requises pour accéder à Amazon Keyspaces, veuillez consulter Exemples de politiques basées sur l'identité Amazon Keyspaces (for Apache Cassandra) dans le Manuel du développeur Amazon Keyspaces (for Apache Cassandra).

Génération d'informations d'identification Amazon Keyspaces (console)

Vous pouvez utiliser le Console de gestion AWS pour générer des informations d'identification Amazon Keyspaces (pour Apache Cassandra) pour vos utilisateurs IAM.

Pour générer des informations d'identification spécifiques au service Amazon Keyspaces (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Users (Utilisateurs), puis choisissez le nom de l'utilisateur ayant besoin d'informations d'identification.
3. Dans l'onglet Security Credentials (informations d'identification de sécurité) sous Credentials for Amazon Keyspaces (informations d'identification pour Amazon Keyspaces) (pour Apache Cassandra) (MCS) sélectionnez Generate credentials (générer des informations d'identification).
4. Vos informations d'identification spécifiques au service sont disponibles. C'est la seule fois que vous pouvez afficher ou télécharger le mot de passe. Vous ne pourrez pas la récupérer plus tard. Cependant, vous pouvez réinitialiser votre mot de passe à tout moment. Enregistrez cet utilisateur et ce mot de passe dans un emplacement sécurisé, car vous en aurez besoin plus tard.

Génération d'informations d'identification Amazon Keyspaces (AWS CLI)

Vous pouvez utiliser le AWS CLI pour générer des informations d'identification Amazon Keyspaces (pour Apache Cassandra) pour vos utilisateurs IAM.

Pour générer des informations d'identification spécifiques au service Amazon Keyspaces (AWS CLI)

- Utilisez la commande suivante :
  - était iam create-service-specific-credential

Génération d'informations d'identification Amazon Keyspaces (API)AWS

Vous pouvez utiliser l' AWS API pour générer des informations d'identification Amazon Keyspaces (pour Apache Cassandra) pour vos utilisateurs IAM.

Pour générer des informations d'identification (API) spécifiques au service Amazon Keyspaces AWS

- Complétez l'opération suivante :
  - CreateServiceSpecificCredential

Trouver les articles non utilisés AWS informations d’identification

Pour renforcer la sécurité de votre compte Compte AWS, supprimez les informations d'identification utilisateur IAM (c'est-à-dire les mots de passe et les clés d'accès) qui ne sont pas nécessaires. Par exemple, lorsque des utilisateurs quittent votre organisation ou n'ont plus besoin d'y AWS accéder, recherchez les informations d'identification qu'ils utilisaient et assurez-vous qu'elles ne sont plus opérationnelles. Idéalement, supprimez les informations d'identification qui ne sont plus requises. Il est toujours possible de les recréer ultérieurement, si nécessaire. Vous devez au moins modifier le mot de passe ou désactiver les clés d'accès afin que les anciens utilisateurs ne soient plus en mesure de s'en servir.

La signification du mot inutilisées peut bien sûr varier : cela indique généralement que les informations d'identification n'ont pas été utilisées au cours d'un laps de temps spécifié.

Recherche de mots de passe inutilisés

Vous pouvez utiliser le Console de gestion AWS pour consulter les informations relatives à l'utilisation des mots de passe par vos utilisateurs. Si vous disposez d'un nombre volumineux d'utilisateurs, vous pouvez utiliser la console pour télécharger un rapport d'informations d'identification contenant des données sur la dernière utilisation du mot de passe de console par chaque utilisateur. Vous pouvez également accéder aux informations à partir de l'API AWS CLI ou de l'API IAM.

Pour rechercher les mots de passe inutilisés (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Au besoin, ajoutez la colonne Console last sign-in (Dernière connexion à la console) à la table des utilisateurs :

   a. Au-dessus de la table à l'extrême droite, choisissez l'icône des paramètres ( ).

   b. Dans Sélectionner les colonnes visibles, sélectionnez Dernière connexion à la console.

   c. Choisissez Confirmer pour revenir à la liste des utilisateurs.
4. La colonne Dernière connexion à la console indique la date à laquelle l'utilisateur s'est connecté pour la dernière fois AWS via la console. Ces informations vous permettent de rechercher les utilisateurs avec mots de passe ne s'étant pas connectés depuis une période donnée. La colonne affiche Jamais pour les utilisateurs avec mots de passe ne s'étant jamais connectés. Aucun indique les utilisateurs sans mot de passe. Les mots de passe qui n'ont pas été utilisés récemment peuvent être supprimés.

   > **Important**
   >
   > En raison d'un problème de service, les données de dernière d'utilisation du mot de passe n'incluent pas l'utilisation du mot de passe entre le 3 mai 2018 et le 23 mai 2018 22:50 14:08 PDT (heure du Pacifique). Cela affecte les dates de dernière connexion affichées dans la console IAM et les dates de dernière utilisation du mot de passe dans le rapport d'identification IAM, et renvoyées par l'opération d'API. GetUser Si des utilisateurs se sont connectés au cours de la période concernée, la date de dernière d'utilisation du mot de passe renvoyée est la date de la dernière connexion de l'utilisateur avant le 3 mai 2018. Pour les utilisateurs qui se sont connectés après le 23 mai 2018 14:08 PDT, la date de dernière utilisation du mot de passe renvoyée est exacte.
   > Si vous utilisez les informations de dernière utilisation du mot de passe pour identifier les informations d'identification non utilisées à supprimer, par exemple en supprimant des utilisateurs qui ne se AWS sont pas connectés au cours des 90 derniers jours, nous vous recommandons d'ajuster votre fenêtre d'évaluation pour inclure les dates postérieures au 23 mai 2018. Par ailleurs, si vos utilisateurs utilisent des clés d'accès pour accéder AWS

par programmation, vous pouvez vous référer aux dernières informations utilisées sur les clés d'accès, car elles sont exactes pour toutes les dates.

Pour rechercher les mots de passe inutilisés en téléchargeant le rapport d'informations d'identification (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, sélectionnez Rapport sur les informations d'identification.
3. Choisissez Télécharger le rapport pour télécharger un fichier CSV nommé `status_reports_<date>T<time>.csv`. La cinquième colonne contient la colonne `password_last_used` avec les dates ou l'une des mentions suivantes :
   - N/A— Utilisateurs auxquels aucun mot de passe n'a été attribué.
   - no_information : utilisateurs n'ayant pas utilisé leur mot de passe depuis le 20 octobre 2014, date à laquelle IAM a commencé le suivi de l'âge des mots de passe.

Pour trouver les mots de passe inutilisés (AWS CLI)

Exécutez la commande suivante pour rechercher les mots de passe inutilisés :

- `aws iam list-users` retourne une liste d'utilisateurs, avec une valeur `PasswordLastUsed` pour chacun. Si la valeur est manquante, cela signifie que l'utilisateur ne dispose pas de mot de passe ou qu'il ne l'a pas utilisé depuis le 20 octobre 2014, date à laquelle IAM a commencé le suivi de l'âge des mots de passe.

Pour trouver les mots de passe inutilisés (AWS API)

Appelez l'opération suivante pour rechercher les mots de passe inutilisés :

- `ListUsers` retourne une collection d'utilisateurs, avec une valeur `<PasswordLastUsed>` pour chacun. Si la valeur est manquante, cela signifie que l'utilisateur ne dispose pas de mot de passe ou qu'il ne l'a pas utilisé depuis le 20 octobre 2014, date à laquelle IAM a commencé le suivi de l'âge des mots de passe.

Pour plus d'informations sur les commandes à utiliser pour le téléchargement du rapport d'informations d'identification, consultez Obtenir des rapports d'identification (AWS CLI).

Recherche des clés d'accès inutilisées

Vous pouvez utiliser le Console de gestion AWS pour consulter les informations d'utilisation des clés d'accès pour vos utilisateurs. Si vous disposez d'un nombre volumineux d'utilisateurs, vous pouvez utiliser la console pour télécharger un rapport d'informations d'identification pour connaître la dernière utilisation des clés d'accès par chaque utilisateur. Vous pouvez également accéder aux informations à partir de l'API AWS CLI ou de l'API IAM.

Pour rechercher les clés d'accès inutilisées (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Utilisateurs.
3. Au besoin, ajoutez la colonne Access key last used (Dernière utilisation de clé d'accès) à la table des utilisateurs :
   a. Au-dessus de la table à l'extrême droite, choisissez l'icône des paramètres ( ).
   b. Dans Sélectionner les colonnes visibles, sélectionnez Dernière clé d'accès utilisée.
   c. Choisissez Confirmer pour revenir à la liste des utilisateurs.
4. La colonne Clé d'accès utilisée pour la dernière fois indique le nombre de jours écoulés depuis le dernier accès par AWS programme de l'utilisateur. Ces informations vous permettent de rechercher les utilisateurs avec des clés d'accès n'ayant pas été utilisées depuis une période donnée. La colonne affiche – pour les utilisateurs sans clés d'accès. Les clés d'accès qui n'ont pas été utilisées récemment peuvent être supprimées.

Pour rechercher les clés d'accès inutilisées en téléchargeant le rapport d'informations d'identification (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, sélectionnez Rapport sur les informations d'identification.

3. Choisissez Télécharger le rapport pour télécharger un fichier CSV nommé `status_reports_<date>T<time>.csv`. Les colonnes 11 à 13 contiennent la date, la région et les informations de service de la dernière utilisation pour la clé d'accès 1. Les colonnes 16 à 18 contiennent les mêmes informations pour la clé d'accès 2. La valeur est N/Asi l'utilisateur ne possède pas de clé d'accès ou s'il ne l'a pas utilisée depuis qu'IAM a commencé à suivre l'âge des clés d'accès le 22 avril 2015.

Pour trouver les clés d'accès non utilisées (AWS CLI)

Exécutez les commandes suivantes pour rechercher les clés d'accès inutilisées :

- `aws iam list-access-keys` retourne des informations sur les clés d'accès d'un utilisateur, y compris l'`AccessKeyID`.
- `aws iam get-access-key-last-used` utilise un ID de clé d'accès et retourne une sortie qui inclut la `LastUsedDate`, la `Region` dans laquelle la clé d'accès a été utilisée la dernière fois et le `ServiceName` du dernier service demandé. Si la valeur de `LastUsedDate` est manquante, ceci indique que la clé d'accès n'a pas été utilisée depuis le 22 avril 2015, date à laquelle IAM a commencé le suivi de l'âge des clés d'accès.

Pour trouver les clés d'accès non utilisées (AWS API)

Appelez les opérations suivantes pour rechercher les clés d'accès inutilisées :

- `ListAccessKeys` retourne une liste de valeurs `AccessKeyID` pour les clés d'accès associées à l'utilisateur spécifié.
- `GetAccessKeyLastUsed` utilise un ID de clé d'accès et retourne une collection de valeurs. Il s'agit notamment de la `LastUsedDate`, de la `Region` dans laquelle la clé d'accès a été utilisée la dernière fois et du `ServiceName` du dernier service demandé. Si la valeur est manquante, cela signifie que l'utilisateur ne dispose pas de clé d'accès ou qu'il ne l'a pas utilisée depuis le 22 avril 2015, date à laquelle IAM a commencé le suivi de l'âge des clés d'accès.

Pour plus d'informations sur les commandes à utiliser pour le téléchargement du rapport d'informations d'identification, consultez Obtenir des rapports d'identification (AWS CLI).

Générez des rapports d'identification pour votre Compte AWS

Vous pouvez générer et télécharger un rapport sur les informations d'identification qui répertorie tous les utilisateurs de votre compte et le statut de leurs diverses informations d'identification, notamment leurs mots de passe, clés d'accès et dispositifs MFA. Vous pouvez obtenir un rapport d'identification à partir des AWS kits de Console de gestion AWS développement logiciel et des outils de ligne de commande, ou de l'API IAM.

> **Note**
>
> Le rapport d'identification IAM inclut uniquement les IAM-managed informations d'identification suivantes : mots de passe, les deux premières clés d'accès par utilisateur, appareils MFA et X.509 certificats de signature. Le rapport n'inclut pas les informations d'identification spécifiques au service (telles que les mots de CodeCommit passe, les clés d'API à long terme Amazon Bedrock ou les clés d'API à long terme Amazon CloudWatch Logs) ni aucune autre clé d'accès utilisateur au-delà des deux premières. Pour une visibilité complète des informations d'identification, utilisez les ListAccessKeys API ListServiceSpecificCredentials et.

Vous pouvez utiliser les rapports sur les informations d'identification à des fins d'audit et de conformité. Vous pouvez utiliser les rapports pour auditer les effets des exigences en matière de cycle de vie des informations d'identification, comme la mise à jour de mot de passe et de clé d'accès. Vous pouvez fournir les rapports à un auditeur externe ou accorder des autorisations à un auditeur pour qu'il télécharge les rapports directement.

Vous pouvez générer un rapport sur les informations d'identification à quelques heures d'intervalle. Lorsque vous demandez un rapport, IAM vérifie d'abord si un rapport Compte AWS a été généré au cours des quatre dernières heures. Le cas échéant, le rapport le plus récent est téléchargé. Si le rapport le plus récent pour le compte a été généré il y a plus de quatre heures ou qu'il n'y a pas eu d'autres rapport généré précédemment pour le compte, IAM génère et télécharge un nouveau rapport.

Rubriques

- Autorisations requises
- Présentation du format du rapport
- Obtention de rapports d'informations d'identification (console)

- Obtenir des rapports d'identification (AWS CLI)
- Obtenir des rapports d'identification (AWS API)

Autorisations requises

Les autorisations suivantes sont nécessaires pour créer et télécharger des rapports :

- Pour créer un rapport sur les informations d'identification : `iam:GenerateCredentialReport`
- Pour télécharger le rapport : `iam:GetCredentialReport`

Présentation du format du rapport

Les rapports sur les informations d'identification sont formatés comme des fichiers CSV (valeurs séparées par une virgule). Vous pouvez ouvrir les fichiers CSV avec des logiciels de feuilles de calcul courants pour effectuer des analyses ou vous pouvez créer une application qui consomme les fichiers CSV par programmation et effectue des analyses personnalisées.

Le fichier CSV contient les colonnes suivantes :

user

Nom convivial de l'utilisateur.

arn

Amazon Resource Name (ARN) de l'utilisateur. Pour plus d'informations sur les ARN, consultez ARN IAM.

user_creation_time

Date et heure de création de l'utilisateur, au format de date et d'heure ISO 8601.

password_enabled

Lorsqu'un utilisateur dispose d'un mot de passe, cette valeur est `TRUE`. Sinon, la valeur est définie comme `FALSE`. Cette valeur est `FALSE` pour les nouveaux comptes membres créés dans le cadre de votre organisation, car ils ne disposent pas des informations d'identification de l'utilisateur racine par défaut.

password_last_used

Date et heure auxquelles le mot de passe de l'utilisateur Utilisateur racine d'un compte AWS ou a été utilisé pour la dernière fois pour se connecter à un AWS site Web, au format date-heure

ISO 8601. AWS les sites Web qui enregistrent l'heure de dernière connexion d'un utilisateur sont Console de gestion AWS les forums de AWS discussion et la AWS place de marché. Quand un mot de passe est utilisé plusieurs fois dans un intervalle de 5 minutes, seule la première utilisation est enregistrée dans ce champ.

- La valeur de ce champ est `no_information` dans les cas suivants :
  - Le mot de passe de l'utilisateur n'a jamais été utilisé.
  - Aucune donnée de connexion n'est associée au mot de passe, comme lorsque le mot de passe de l'utilisateur n'a pas été utilisé depuis le 20 octobre 2014, date de début du suivi de cette information par IAM.
- La valeur de ce champ est `N/A` (non applicable) lorsque l'utilisateur ne dispose pas de mot de passe.

> **Important**
>
> En raison d'un problème de service, les données de dernière d'utilisation du mot de passe n'incluent pas l'utilisation du mot de passe entre le 3 mai 2018 et le 23 mai 2018 22:50 14:08 PDT (heure du Pacifique). Cela affecte les dates de dernière connexion affichées dans la console IAM et les dates de dernière utilisation du mot de passe dans le rapport d'identification IAM et renvoyées par l'opération d'API. GetUser Si des utilisateurs se sont connectés au cours de la période concernée, la date de dernière d'utilisation du mot de passe renvoyée est la date de la dernière connexion de l'utilisateur avant le 3 mai 2018. Pour les utilisateurs qui se sont connectés après le 23 mai 2018 14:08 PDT, la date de dernière utilisation du mot de passe renvoyée est exacte.
> Si vous utilisez les informations du dernier mot de passe utilisé pour identifier les informations d'identification non utilisées à supprimer, par exemple en supprimant des utilisateurs qui ne se sont pas connectés au AWS cours des 90 derniers jours, nous vous recommandons d'ajuster votre fenêtre d'évaluation pour inclure des dates postérieures au 23 mai 2018. Sinon, si vos utilisateurs utilisent des clés d'accès pour accéder AWS par programmation, vous pouvez vous référer aux informations relatives à la dernière clé d'accès utilisée, car elles sont exactes pour toutes les dates.

password_last_changed

Date et heure de la dernière définition du mot de passe de l'utilisateur, au format de date et d'heure ISO 8601. Si l'utilisateur ne dispose pas d'un mot de passe, la valeur de ce champ est `N/A` (non applicable).

password_next_rotation

Lorsque le compte dispose d'une politique de mot de passe qui nécessite la rotation du mot de passe, ce champ contient la date et l'heure, au format de date et d'heure ISO 8601, auxquelles l'utilisateur doit définir un nouveau mot de passe. La valeur de Compte AWS (racine) est toujours`not_supported`.

mfa_active

Lorsqu'un périphérique d'authentification multi-facteur (MFA) a été activé pour l'utilisateur, cette valeur est `TRUE`. Sinon, la valeur est définie comme `FALSE`.

access_key_1_active

Lorsque l'utilisateur dispose d'une clé d'accès et que l'état de celle-ci est `Active`, cette valeur est `TRUE`. Sinon, la valeur est définie comme `FALSE`. S'applique à la fois à l'utilisateur racine du compte et aux utilisateurs IAM.

access_key_1_last_rotated

Date et heure, au format de date et d'heure ISO 8601, auxquelles la clé d'accès de l'utilisateur a été créée ou modifiée pour la dernière fois. Si l'utilisateur ne dispose pas d'une clé d'accès active, la valeur de ce champ est `N/A` (non applicable). S'applique à la fois à l'utilisateur racine du compte et aux utilisateurs IAM.

access_key_1_last_used_date

Date et heure, au format de date et d'heure ISO 8601, auxquelles la clé d'accès de l'utilisateur a été utilisée pour la dernière fois pour se connecter à une demande d'API AWS . Quand une clé d'accès est utilisée plusieurs fois dans un intervalle de 15 minutes, seule la première utilisation est enregistrée dans ce champ. S'applique à la fois à l'utilisateur racine du compte et aux utilisateurs IAM.

La valeur de ce champ est `N/A` (non applicable) dans les cas suivants :

- L'utilisateur ne dispose pas d'une clé d'accès.
- La clé d'accès n'a jamais été utilisée.

- La clé d'accès n'a pas été utilisée depuis le 22 avril 2015, date de début du suivi de cette information par IAM.

access_key_1_last_used_region

Région AWS dans laquelle la clé d'accès a été utilisée pour la dernière fois. Quand une clé d'accès est utilisée plusieurs fois dans un intervalle de 15 minutes, seule la première utilisation est enregistrée dans ce champ. S'applique à la fois à l'utilisateur racine du compte et aux utilisateurs IAM.

La valeur de ce champ est `N/A` (non applicable) dans les cas suivants :

- L'utilisateur ne dispose pas d'une clé d'accès.
- La clé d'accès n'a jamais été utilisée.
- La clé d'accès a été utilisée pour la dernière fois avant le 22 avril 2015, date de début du suivi de cette information par IAM.
- Le dernier service utilisé ne l'est pas Region-specific, par exemple Amazon S3.

access_key_1_last_used_service

AWS Service auquel vous avez accédé le plus récemment à l'aide de la clé d'accès. La valeur de ce champ utilise l'espace de noms du service (par exemple, `s3` pour Amazon S3 et `ec2` pour Amazon EC2). Quand une clé d'accès est utilisée plusieurs fois dans un intervalle de 15 minutes, seule la première utilisation est enregistrée dans ce champ. S'applique à la fois à l'utilisateur racine du compte et aux utilisateurs IAM.

La valeur de ce champ est `N/A` (non applicable) dans les cas suivants :

- L'utilisateur ne dispose pas d'une clé d'accès.
- La clé d'accès n'a jamais été utilisée.
- La clé d'accès a été utilisée pour la dernière fois avant le 22 avril 2015, date de début du suivi de cette information par IAM.

access_key_2_active

Lorsque l'utilisateur dispose d'une seconde clé d'accès et que l'état de celle-ci est `Active`, cette valeur est `TRUE`. Sinon, la valeur est définie comme `FALSE`. S'applique à la fois à l'utilisateur racine du compte et aux utilisateurs IAM.

> **Note**
>
> Les utilisateurs peuvent avoir jusqu'à deux clés d'accès, afin de faciliter la rotation en mettant d'abord à jour la clé, puis en supprimant la clé précédente. Pour plus d'informations sur la mise à jour des clés d'accès, veuillez consulter Mise à jour des clés d'accès.

access_key_2_last_rotated

Date et heure, au format de date et d'heure ISO 8601, de la création ou de la dernière mise à jour de la deuxième clé d'accès de l'utilisateur. Si l'utilisateur ne dispose pas d'une seconde clé d'accès active, la valeur de ce champ est `N/A` (non applicable). S'applique à la fois à l'utilisateur racine du compte et aux utilisateurs IAM.

access_key_2_last_used_date

Date et heure, au format date-heure ISO 8601, auxquelles la deuxième clé d'accès de l'utilisateur a été utilisée pour la dernière fois pour signer une AWS demande d'API. Quand une clé d'accès est utilisée plusieurs fois dans un intervalle de 15 minutes, seule la première utilisation est enregistrée dans ce champ. S'applique à la fois à l'utilisateur racine du compte et aux utilisateurs IAM.

La valeur de ce champ est `N/A` (non applicable) dans les cas suivants :

- L'utilisateur ne dispose pas d'une seconde clé d'accès.
- La seconde clé d'accès de l'utilisateur n'a jamais été utilisée.
- La seconde clé d'accès de l'utilisateur a été utilisée pour la dernière fois avant le 22 avril 2015, date de début du suivi de cette information par IAM.

access_key_2_last_used_region

Région AWS dans laquelle la seconde clé d'accès de l'utilisateur a été utilisée pour la dernière fois. Quand une clé d'accès est utilisée plusieurs fois dans un intervalle de 15 minutes, seule la première utilisation est enregistrée dans ce champ. S'applique à la fois à l'utilisateur racine du compte et aux utilisateurs IAM. La valeur de ce champ est `N/A` (non applicable) dans les cas suivants :

- L'utilisateur ne dispose pas d'une seconde clé d'accès.
- La seconde clé d'accès de l'utilisateur n'a jamais été utilisée.

- La seconde clé d'accès de l'utilisateur a été utilisée pour la dernière fois avant le 22 avril 2015, date de début du suivi de cette information par IAM.
- Le dernier service utilisé ne l'est pas Region-specific, par exemple Amazon S3.

access_key_2_last_used_service

AWS Service auquel l'utilisateur a accédé pour la dernière fois à l'aide de la deuxième clé d'accès. La valeur de ce champ utilise l'espace de noms du service (par exemple, `s3` pour Amazon S3 et `ec2` pour Amazon EC2). Quand une clé d'accès est utilisée plusieurs fois dans un intervalle de 15 minutes, seule la première utilisation est enregistrée dans ce champ. S'applique à la fois à l'utilisateur racine du compte et aux utilisateurs IAM. La valeur de ce champ est `N/A` (non applicable) dans les cas suivants :

- L'utilisateur ne dispose pas d'une seconde clé d'accès.
- La seconde clé d'accès de l'utilisateur n'a jamais été utilisée.
- La seconde clé d'accès de l'utilisateur a été utilisée pour la dernière fois avant le 22 avril 2015, date de début du suivi de cette information par IAM.

cert_1_active

Lorsque l'utilisateur possède un certificat de X.509 signature et que le statut de ce certificat est`Active`, cette valeur est`TRUE`. Sinon, la valeur est définie comme `FALSE`.

cert_1_last_rotated

Date et heure, au format de date et d'heure ISO 8601, auxquelles le certificat de signature de l'utilisateur a été créé ou modifié pour la dernière fois. Si l'utilisateur ne dispose pas d'un certificat de signature actif, la valeur de ce champ est `N/A` (non applicable).

cert_2_active

Lorsque l'utilisateur possède un deuxième certificat de X.509 signature et que le statut de ce certificat est`Active`, cette valeur est`TRUE`. Sinon, la valeur est définie comme `FALSE`.

> **Note**
>
> Les utilisateurs peuvent avoir jusqu'à deux certificats de X.509 signature, afin de faciliter la rotation des certificats.

cert_2_last_rotated

Date et heure, au format de date et d'heure ISO 8601, auxquelles le second certificat de signature de l'utilisateur a été créé ou modifié pour la dernière fois. Si l'utilisateur ne dispose pas d'un second certificat de signature actif, la valeur de ce champ est N/A (non applicable).

information_d'identifications_supplémentaires

Lorsque l'utilisateur possède plus de deux clés d'accès ou certificats, cette valeur correspond au nombre de clés d'accès ou de certificats supplémentaires et aux actions que vous pouvez utiliser pour répertorier les clés d'accès ou les certificats associés à l'utilisateur.

Obtention de rapports d'informations d'identification (console)

Vous pouvez utiliser le Console de gestion AWS pour télécharger un rapport d'identification sous forme de fichier de valeurs séparées par des virgules (CSV).

Pour télécharger un rapport sur les informations d'identification (console)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation, sélectionnez Rapport sur les informations d'identification.
3. Choisissez Télécharger le rapport.

Obtenir des rapports d'identification (AWS CLI)

Pour télécharger un rapport sur les informations d'identification (AWS CLI)

1. Générez un rapport sur les informations d'identification. AWS stocke un seul rapport. Si un rapport existe, la génération d'un rapport sur les informations d'identification remplace le rapport précédent. `aws iam generate-credential-report`
2. Affichez le dernier rapport généré : `aws iam get-credential-report`

Obtenir des rapports d'identification (AWS API)

Pour télécharger un rapport sur les informations d'identification (AWS API)

1. Générez un rapport sur les informations d'identification. AWS stocke un seul rapport. Si un rapport existe, la génération d'un rapport sur les informations d'identification remplace le rapport précédent. `GenerateCredentialReport`
2. Affichez le dernier rapport généré : `GetCredentialReport`

Informations d'identification IAM pour CodeCommit : informations d'identification Git, clés SSH et AWS clés d'accès

CodeCommit est un service de contrôle de version géré qui héberge des référentiels Git privés dans le AWS cloud. Pour l'utiliser CodeCommit, vous devez configurer votre client Git pour qu'il communique avec CodeCommit les référentiels. Dans le cadre de cette configuration, vous fournissez des informations d'identification IAM qui CodeCommit peuvent être utilisées pour vous authentifier. IAM prend en charge CodeCommit trois types d'informations d'identification :

- Informations d'identification Git, paire nom d'utilisateur et mot de passe générée par IAM que vous pouvez utiliser pour communiquer avec les CodeCommit référentiels via HTTPS.
- Les clés SSH sont une paire de clés publique-privée générée localement que vous pouvez associer à votre utilisateur IAM pour communiquer avec les CodeCommit référentiels via SSH.
- AWS clés d'accès, que vous pouvez utiliser avec l'assistant d'identification inclus dans le AWS CLI pour communiquer avec les CodeCommit référentiels via HTTPS.

> **Note**
>
> Vous ne pouvez pas utiliser les clés SSH ni les informations d'identification Git pour accéder aux référentiels dans un autre compte AWS . Pour savoir comment configurer l'accès aux CodeCommit référentiels pour les utilisateurs et les groupes IAM d'un autre utilisateur Compte AWS, voir Configurer l'accès entre comptes à un AWS CodeCommit référentiel à l'aide de rôles dans le Guide de l'AWS CodeCommit utilisateur.

Pour plus d'informations sur chaque option, consultez les sections suivantes.

Utiliser les informations d'identification Git et HTTPS avec CodeCommit (recommandé)

Avec les informations d'identification Git, vous générez une paire mot de passe-nom utilisateur statique pour votre utilisateur IAM, puis utilisez ces informations d'identification pour les connexions HTTPS. Vous pouvez également utiliser ces informations d'identification avec n'importe quel outil tiers ou environnement de développement intégré (IDE) prenant en charge les informations d'identification Git statiques.

Etant donné que ces informations d'identification sont universelles pour tous les systèmes d'exploitation pris en charge et compatibles avec la plupart des systèmes de gestion des informations d'identification, environnements de développement et autres outils de développement logiciel, c'est la méthode recommandée. Vous pouvez réinitialiser le mot de passe pour les informations d'identification Git à tout moment. Vous pouvez également désactiver les informations d'identification ou les supprimer si vous n'en avez plus besoin.

> **Note**
>
> Pour les informations d'identification Git, vous ne pouvez pas choisir vous-même votre nom d'utilisateur et votre mot de passe. IAM génère ces informations d'identification pour vous aider à garantir qu'elles répondent aux normes de sécurité AWS et qu'elles sécurisent les référentiels dans. CodeCommit Vous pouvez télécharger les informations d'identification une seule fois, au moment elles sont générées. Veillez à enregistrer les informations d'identification dans un emplacement sûr. Si nécessaire, vous pouvez réinitialiser le mot de passe à tout moment, mais cela invalidera les connexions configurées avec l'ancien mot de passe. Vous devez reconfigurer les connexions pour qu'elles utilisent le nouveau mot de passe avant de vous connecter.

Pour plus d'informations, consultez les rubriques suivantes :

- Pour créer un utilisateur IAM, veuillez consulter Créez un utilisateur IAM dans votre Compte AWS.
- Pour générer et utiliser des informations d'identification Git avec CodeCommit, consultez la section Pour les utilisateurs HTTPS utilisant des informations d'identification Git dans le guide de AWS CodeCommit l'utilisateur.

> **Note**
>
> Le fait de modifier le nom d'un utilisateur IAM après avoir généré des informations d'identification Git ne modifie pas le nom utilisateur des informations d'identification Git. Le nom utilisateur et le mot de passe restent identiques et sont toujours valides.

Pour mettre à jour les informations d'identification spécifiques au service

1. Créez un second ensemble d'informations d'identification spécifiques au service en plus de l'ensemble actuellement en cours d'utilisation.
2. Mettez à jour toutes vos applications afin qu'elles utilisent le nouvel ensemble d'informations d'identification et vérifiez que les applications fonctionnent.
3. Modifiez l'état des informations d'identification d'origine et utilisez Inactif.
4. Vérifiez que toutes vos applications fonctionnent toujours.
5. Supprimer les informations d'identification spécifiques au service inactives.

Utilisez des clés SSH et SSH avec CodeCommit

Avec les connexions SSH, vous créez des fichiers de clés publiques et privées sur votre machine locale que Git CodeCommit utilise pour l'authentification SSH. Vous associez la clé publique à votre utilisateur IAM et stockez la clé privée sur votre ordinateur local. Pour plus d'informations, consultez les rubriques suivantes :

- Pour créer un utilisateur IAM, veuillez consulter Créez un utilisateur IAM dans votre Compte AWS.
- Pour créer une clé publique SSH et l'associer à un utilisateur IAM, consultez Pour les connexions SSH sous Linux, macOS ou Unix ou Pour des connexions SSH sous Windows dans le Guide de l'utilisateur AWS CodeCommit .

> **Note**
>
> La clé publique doit être codée au format ssh-rsa ou PEM. Le longueur binaire minimale de la clé publique est de 2048 bits et la longueur maximale de 16 384 bits. Cette longueur est distincte de la taille du fichier que vous chargez. Par exemple, vous pouvez générer une clé de 2048 bits et le fichier PEM résultant a une longueur de 1679 octets. Si vous

fournissez votre clé publique dans un autre format ou une autre taille, un message d'erreur vous indiquera que le format de clé n'est pas valide.

Utilisez HTTPS avec l'assistant AWS CLI d'identification et CodeCommit

Comme alternative aux connexions HTTPS avec des informations d'identification Git, vous pouvez autoriser Git à utiliser une version signée cryptographiquement de vos informations d'identification utilisateur IAM ou de votre rôle d'instance Amazon EC2 chaque fois que Git doit s'authentifier pour interagir AWS avec des référentiels. CodeCommit Il s'agit de la seule méthode de connexion pour CodeCommit les référentiels qui ne nécessite pas d'utilisateur IAM. C'est également la seule méthode qui fonctionne avec un accès fédéré et des informations d'identification temporaires. Pour plus d'informations, consultez les rubriques suivantes :

- Pour en savoir plus sur l'accès fédéré, consultez Fournisseurs d'identité et fédération au sein AWS et Accès aux utilisateurs authentifiés en externe (fédération d'identité).
- Pour en savoir plus sur les informations d'identification temporaires, consultez Informations d'identification de sécurité temporaires dans IAM et Accès temporaire à des référentiels CodeCommit.

L'assistant AWS CLI d'identification n'est pas compatible avec les autres systèmes d'assistance aux identifiants, tels que Keychain Access ou Windows Credential Management. Il existe d'autres considérations relatives à la configuration lorsque vous configurez des connexions HTTPS à l'aide de l'assistant d'informations d'identification. Pour plus d'informations, voir Pour les connexions HTTPS sous Linux, macOS ou Unix avec l'assistant AWS CLI d'identification ou Connexions HTTPS sous Windows avec l'assistant AWS CLI d'identification dans le guide de l'utilisateur.AWS CodeCommit

Gestion des certificats de serveur dans IAM

Pour activer les connexions HTTPS à votre site Web ou à votre application dans AWS, vous avez besoin d'un certificat de SSL/TLS serveur. Pour les certificats dans une région prise en charge par AWS Certificate Manager (ACM), nous vous recommandons d'utiliser ACM pour allouer, gérer et déployer vos certificats de serveur. Dans les régions non prises en charge, vous devez utiliser IAM en tant que gestionnaire de certificats. Pour savoir quelles régions sont prises en charge par ACM, consultez Points de terminaison et quotas AWS Certificate Manager (français non garanti) dans Références générales AWS.

> **Important**
>
> ACM est l'outil préféré pour mettre en service, gérer et déployer vos certificats de serveur. Avec ACM, vous pouvez demander un certificat ou déployer un certificat ACM existant ou externe sur des ressources. AWS Les certificats fournis par ACM sont gratuits et renouvelés automatiquement. Dans une région prise en charge, vous pouvez utiliser ACM pour gérer des certificats de serveur depuis la console ou par programmation. Pour plus d'informations sur l'utilisation d'ACM, consultez le Guide de l'utilisateur AWS Certificate Manager . Pour plus d'informations sur la demande d'un certificat ACM, consultez Demander un certificat public ou Demander in certificat privé dans le Guide de l'utilisateur AWS Certificate Manager . Pour plus d'informations sur l'importation de certificats tiers dans ACM, consultez Importation de certificats dans le Guide de l'utilisateur AWS Certificate Manager .

Utilisez IAM comme gestionnaire de certificats uniquement lorsque vous devez prendre en charge des connexions HTTPS dans une région non prise en charge par ACM. IAM chiffre en toute sécurité vos clés privées et stocke la version chiffrée dans le magasin de certificats SSL IAM. IAM prend en charge le déploiement de certificats de serveur dans toutes les régions, mais vous devez obtenir votre certificat auprès d'un fournisseur externe pour pouvoir l'utiliser. AWS Vous ne pouvez pas télécharger de certificat ACM dans IAM. De plus, vous ne pouvez pas gérer vos certificats depuis la console IAM.

Pour plus d'informations sur le chargement de certificats tiers dans IAM, consultez les rubriques suivantes.

Rubriques

- Téléchargez un certificat de serveur (AWS API)
- AWS Opérations d'API pour les certificats de serveur
- Résolution des problèmes liés aux certificats de serveur

Téléchargez un certificat de serveur (AWS API)

Pour télécharger un certificat de serveur dans IAM, vous devez fournir le certificat et sa clé privée correspondante. Si le certificat n'est pas auto-signé, vous devez également fournir une chaîne de certificats. (Vous n'avez pas besoin d'une chaîne de certificats lorsque vous chargez un certificat auto-signé.) Avant de charger un certificat, vérifiez que vous avez tous ces éléments et qu'ils répondent aux critères suivants :

- Le certificat doit être valide au moment du chargement. Vous ne pouvez pas charger un certificat avant le début de sa période de validité (date `NotBefore` du certificat) ou après son expiration (date `NotAfter` du certificat).
- La clé privée doit être non chiffrée. Vous ne pouvez pas charger une clé privée qui est protégée par un mot de passe ou une phrase passe. Pour obtenir de l'aide pour déchiffrer une clé privée chiffrée, consultez Résolution des problèmes liés aux certificats de serveur.
- Le certificat, la clé privée et la chaîne de certificats doivent tous l'être PEM-encoded. Pour obtenir de l'aide pour convertir ces éléments au format PEM, consultez Résolution des problèmes liés aux certificats de serveur.

Pour utiliser l'API IAM afin de télécharger un certificat, envoyez une UploadServerCertificate demande. L'exemple suivant montre comment procéder avec l'AWS Command Line Interface (AWS CLI). Dans cet exemple il est supposé que :

- Le PEM-encoded certificat est stocké dans un fichier nommé`Certificate.pem`.
- La chaîne de PEM-encoded certificats est stockée dans un fichier nommé`CertificateChain.pem`.
- La PEM-encoded clé privée non chiffrée est stockée dans un fichier nommé`PrivateKey.pem`.
- (Facultatif) Vous souhaitez baliser le certificat de serveur avec une paire clé-valeur. Par exemple, vous pouvez ajouter la clé de balise `Department` et la valeur de balise `Engineering` pour vous aider à identifier et organiser vos certificats.

Pour utiliser l'exemple de commande suivant, remplacez ces noms de fichier par les vôtres. Remplacez-le *`ExampleCertificate`* par un nom pour le certificat que vous avez chargé. Si vous souhaitez baliser le certificat, remplacez la paire clé-valeur *`ExampleKey`* et *`ExampleValue`* balisez par vos propres valeurs. Tapez la commande sur une seule ligne continue. L'exemple suivant inclut des sauts de ligne et des espaces supplémentaires pour le rendre plus facile à lire.

```
aws iam upload-server-certificate --server-certificate-name ExampleCertificate
                                  --certificate-body file://Certificate.pem
                                  --certificate-chain file://CertificateChain.pem
                                  --private-key file://PrivateKey.pem
                                  --tags '{"Key": "ExampleKey", "Value":
 "ExampleValue"}'
```

Lorsque la commande précédente aboutit, elle renvoie des métadonnées relatives au certificat chargé, y compris son Amazon Resource Name (ARN), son nom convivial, son identifiant (ID), sa date d'expiration, ses balises, etc.

> **Note**
>
> Si vous chargez un certificat de serveur à utiliser avec Amazon CloudFront, vous devez spécifier un chemin à l'aide de `--path` cette option. Le chemin doit commencer par `/cloudfront` et doit inclure une barre oblique de fin (par exemple, `/cloudfront/test/`).

Pour utiliser le AWS Tools for Windows PowerShell pour télécharger un certificat, utilisez Publish-IAMServerCertificate.

AWS Opérations d'API pour les certificats de serveur

Utilisez les commandes suivantes pour afficher, étiqueter, renommer et supprimer les certificats de serveur.

- GetServerCertificateUtilisez-le pour récupérer un certificat. Cette requête renvoie le certificat, la chaîne de certificats (si une chaîne de certificats a été chargée) et des métadonnées relatives au certificat.

  > **Note**
  >
  > Vous ne pouvez pas télécharger ou récupérer une clé privée depuis IAM après l'avoir téléchargée.

- Get-IAMServerCertificateUtilisez-le pour récupérer un certificat.
- Utilisez-le ListServerCertificates pour répertorier les certificats de serveur que vous avez chargés. La requête renvoie une liste qui contient les métadonnées relatives à chaque certificat.
- Utilisez-le Get-IAMServerCertificates pour répertorier les certificats de serveur que vous avez chargés.
- Permet TagServerCertificate de baliser un certificat de serveur existant.
- Permet d'UntagServerCertificateannuler le balisage d'un certificat de serveur.
- Permet UpdateServerCertificate de renommer un certificat de serveur ou de mettre à jour son chemin.

L'exemple suivant montre comment procéder avec l'interface AWS CLI.

Pour utiliser la commande suivante, remplacez les anciens et nouveaux noms de certificat et le chemin d'accès du certificat, et tapez la commande sur une seule ligne continue. L'exemple suivant inclut des sauts de ligne et des espaces supplémentaires pour le rendre plus facile à lire.

```
aws iam update-server-certificate --server-certificate-name ExampleCertificate
                                  --new-server-certificate-
name CloudFrontCertificate
                                  --new-path /cloudfront/
```

AWS Tools for Windows PowerShell Pour renommer un certificat de serveur ou mettre à jour son chemin d'accès, utilisez Update-IAMServerCertificate.

- Permet DeleteServerCertificate de supprimer un certificat de serveur.

  Pour utiliser le AWS Tools for Windows PowerShell pour supprimer un certificat de serveur, utilisez Remove-IAMServerCertificate.

Résolution des problèmes liés aux certificats de serveur

Avant de télécharger un certificat vers IAM, vous devez vous assurer que le certificat, la clé privée et la chaîne de certificats sont tous PEM-encoded présents. Vous devez également vous assurer que la clé privée est non chiffrée. Voir les exemples suivantes.

Example Exemple de PEM-encoded certificat

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

Example Exemple de PEM-encoded clé privée non chiffrée

```
-----BEGIN RSA PRIVATE KEY-----
Base64-encoded private key
-----END RSA PRIVATE KEY-----
```

Example Exemple de chaîne PEM-encoded de certificats

Une chaîne de certificats contient un ou plusieurs certificats. Vous pouvez utiliser un éditeur de texte, la commande copy sous Windows ou la commande cat Linux pour concaténer vos fichiers de certificats dans une chaîne. Lorsque vous incluez plusieurs certificats, chacun d'entre eux doit approuver le certificat précédent. Vous pouvez effectuer cette opération en concaténant les certificats, y compris le certificat d'autorité de certification racine en dernier.

L'exemple suivant contient trois certificats, mais votre chaîne de certificats peut en contenir plus ou moins.

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

Si ces éléments ne sont pas au bon format pour le téléchargement vers IAM, vous pouvez utiliser OpenSSL pour les convertir au format approprié.

Pour convertir un certificat ou une chaîne de certificats DER en PEM

Utilisez la commande OpenSSL x509, comme dans l'exemple suivant. Dans l'exemple de commande suivant, remplacez *Certificate.der* par le nom du fichier contenant votre DER-encoded certificat. *Certificate.pem*Remplacez-le par le nom préféré du fichier de sortie qui doit contenir le PEM-encoded certificat.

```
openssl x509 -inform DER -in Certificate.der -outform PEM -out Certificate.pem
```

Pour convertir une clé privée de DER en PEM

Utilisez la commande OpenSSL rsa, comme dans l'exemple suivant. Dans l'exemple de commande suivant, remplacez *PrivateKey.der* par le nom du fichier contenant votre clé DER-encoded privée. *PrivateKey.pem*Remplacez-le par le nom préféré du fichier de sortie contenant la clé PEM-encoded privée.

```
openssl rsa -inform DER -in PrivateKey.der -outform PEM -out PrivateKey.pem
```

Pour déchiffrer une clé privée chiffrée (supprimer le mot de passe ou la phrase de passe)

Utilisez la commande OpenSSL rsa, comme dans l'exemple suivant. Pour utiliser la commande suivante, remplacez *EncryptedPrivateKey.pem* par le nom du fichier qui contient votre clé privée chiffrée. *PrivateKey.pem*Remplacez-le par le nom préféré du fichier de sortie contenant la clé privée PEM-encoded non chiffrée.

```
openssl rsa -in EncryptedPrivateKey.pem -out PrivateKey.pem
```

Pour convertir un ensemble de certificats de PKCS#12 (PFX) en PEM

Utilisez la commande OpenSSL pkcs12, comme dans l'exemple suivant. Dans la commande suivante, remplacez *CertificateBundle.p12* par le nom du fichier qui contient votre ensemble de certificats codé PKCS#12. *CertificateBundle.pem*Remplacez-le par le nom préféré du fichier de sortie contenant le bundle de PEM-encoded certificats.

```
openssl pkcs12 -in CertificateBundle.p12 -out CertificateBundle.pem -nodes
```

Pour convertir un ensemble de certificats de PKCS#7 en PEM

Utilisez la commande OpenSSL pkcs7, comme dans l'exemple suivant. Dans la commande suivante, remplacez *CertificateBundle.p7b* par le nom du fichier qui contient votre ensemble de certificats codé PKCS#7. *CertificateBundle.pem*Remplacez-le par le nom préféré du fichier de sortie contenant le bundle de PEM-encoded certificats.

```
openssl pkcs7 -in CertificateBundle.p7b -print_certs -out CertificateBundle.pem
```

Groupes d'utilisateurs IAM

Un groupe d'utilisateurs IAM est un ensemble d'utilisateurs IAM. Les groupes d'utilisateurs vous permettent de spécifier des autorisations pour plusieurs utilisateurs, ce qui permet de gérer plus

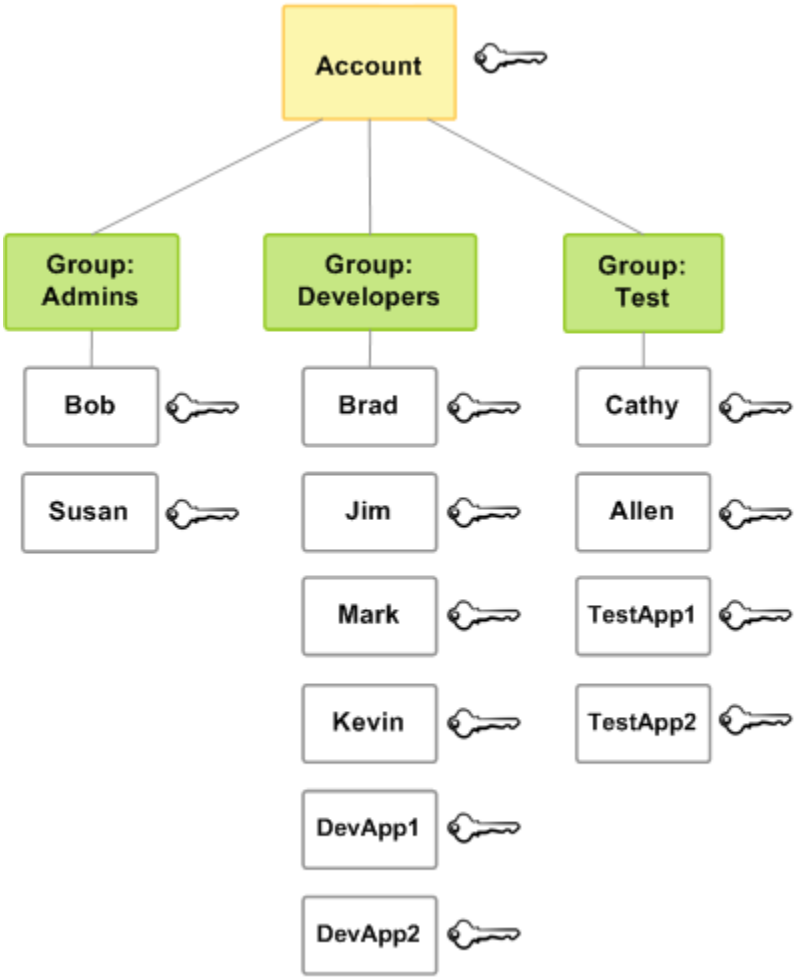
facilement les autorisations pour ces utilisateurs. Par exemple, vous pouvez avoir un groupe d'utilisateurs appelé Admins et accorder à ce groupe d'utilisateurs des autorisations d'administrateur typiques. Tous les utilisateurs de ce groupe d'utilisateurs reçoivent automatiquement les autorisations de groupe Admins. Si un nouvel utilisateur rejoint votre organisation et a besoin de privilèges d'administrateur, vous pouvez lui attribuer les autorisations appropriées en l'ajoutant au groupe d'utilisateurs Admins. Si une personne change de poste dans votre organisation, au lieu de modifier les autorisations de cet utilisateur, vous pouvez retirer celui-ci de l'ancien groupe IAM et l'ajouter aux nouveaux groupes IAM appropriés.

Vous pouvez attacher une politique basée sur l'identité à un groupe d'utilisateurs afin que tous les utilisateurs de ce groupe reçoivent les autorisations de la politique. Vous ne pouvez pas identifier un groupe d'utilisateurs en tant que `Principal` dans une politique (telle qu'une politique basée sur les ressources) car les groupes se rapportent aux autorisations, et non à l'authentification, et les principaux sont des entités IAM authentifiées. Pour de plus amples informations sur les types de politiques, veuillez consulter Identity-based politiques et politiques basées sur les ressources.

Voici quelques caractéristiques importantes des groupes IAM :

- Un groupe d'utilisateurs peut contenir de nombreux utilisateurs et un utilisateur peut appartenir à plusieurs groupes d'utilisateurs.
- Les groupes IAM ne peuvent pas être imbriqués ; ils ne peuvent contenir que des utilisateurs, pas d'autres groupes IAM.
- Il n'existe pas de groupe d'utilisateurs par défaut qui inclut automatiquement tous les utilisateurs de l' Compte AWS. Si vous souhaitez disposer d'un groupe d'utilisateurs de ce type, vous devez le créer et lui affecter chaque nouvel utilisateur.
- Le nombre et la taille des ressources IAM présentes dans un Compte AWS, tels que le nombre de groupes et le nombre de groupes auxquels un utilisateur peut appartenir, sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

Le schéma suivant illustre un exemple simple de petite entreprise. Le propriétaire de l'entreprise crée un groupe d'utilisateurs `Admins` pour que des utilisateurs créent et gèrent d'autres utilisateurs au fur et à mesure que l'entreprise se développe. Le groupe d'utilisateurs `Admins` crée un groupe d'utilisateurs `Developers` et un groupe d'utilisateurs `Test`. Chacun de ces groupes IAM est composé d'utilisateurs (humains et applications) qui interagissent avec AWS (Jim, Brad, DevApp 1, etc.). Chaque utilisateur dispose d'un ensemble individuel d'informations d'identification de sécurité. Dans cet exemple, chaque utilisateur appartient à un seul groupe d'utilisateurs. Cependant, les utilisateurs peuvent appartenir à plusieurs groupes IAM.



Après avoir créé un groupe d'utilisateurs, vous pouvez Afficher les groupes IAMAttachement d'une politique à un groupe d'utilisateurs IAM, etAffectation d'un nouveau nom à un groupe d'utilisateurs IAM.

Création de groupes IAM

> **Note**
>
> En tant que bonne pratique, nous vous recommandons de demander aux utilisateurs humains d'utiliser la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires. Si vous suivez les bonnes pratiques, vous ne gérez pas les utilisateurs et les groupes IAM. Au lieu de cela, vos utilisateurs et vos groupes sont gérés en dehors de l'extérieur AWS et peuvent accéder aux AWS ressources en tant qu'identité fédérée. Une identité fédérée est un utilisateur de l'annuaire des utilisateurs

de votre entreprise, un fournisseur d'identité Web, le AWS Directory Service, le répertoire Identity Center ou tout utilisateur qui accède AWS aux services à l'aide des informations d'identification fournies par le biais d'une source d'identité. Les identités fédérées utilisent les groupes définis par leur fournisseur d'identité. Si vous en utilisez AWS IAM Identity Center, consultez la section Gérer les identités dans IAM Identity Center dans le Guide de l'AWS IAM Identity Center utilisateur pour plus d'informations sur la création d'utilisateurs et de groupes dans IAM Identity Center.

Vous créez des groupes IAM pour gérer les autorisations d'accès de plusieurs utilisateurs ayant des rôles ou des responsabilités similaires. En associant des politiques à ces groupes, vous pouvez accorder ou révoquer des autorisations pour des ensembles complets d'utilisateurs. Cela simplifie la gestion de vos politiques de sécurité, car les modifications que vous apportez aux autorisations d'un groupe sont automatiquement appliquées à tous les membres de ce groupe, garantissant ainsi un contrôle d'accès cohérent. Après avoir créé le groupe, attribuez-lui des autorisations en fonction du type de tâches que vous prévoyez de confier aux utilisateurs IAM qui en feront partie, puis ajoutez les utilisateurs IAM au groupe.

Pour obtenir plus d'informations sur les autorisations nécessaires pour créer un groupe IAM, consultez Autorisations requises pour accéder aux autres ressources IAM.

Pour créer un groupe IAM et attacher des politiques

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), puis Create group (Créer un groupe).
3. Pour User group name (Nom du groupe d'utilisateurs), saisissez le nom du groupe.

> **Note**
>
> Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas. Les noms de groupe peuvent combiner jusqu'à 128 lettres, chiffres et caractères suivants : plus (+), égal (=), virgule (,), point (.), arobase (@), trait de soulignement (_) et tiret (-). Les

noms doivent être uniques au sein d'un compte. Ils ne sont pas sensibles à la casse. Par exemple, vous ne pouvez pas créer deux groupes nommés **`ADMINS`** et **`admins`**.

4. Dans la liste des utilisateurs, cochez la case en regard de chacun des utilisateurs à ajouter au groupe.
5. Dans la liste de politiques, activez la case à cocher en regard de chaque politique devant s'appliquer à tous les membres du groupe.
6. Choisissez Créer un groupe.

AWS CLI

Exécutez la commande suivante :

- aws iam create-group

API

Appelez l'opération suivante :

- CreateGroup

Afficher les groupes IAM

Vous pouvez répertorier tous les groupes IAM de votre compte, les utilisateurs d'un groupe IAM ou les groupes IAM auxquels appartient un utilisateur. Si vous utilisez la CLI ou l'API, vous pouvez afficher tous les groupes IAM avec un préfixe de chemin d'accès spécifique.

Console

Pour répertorier tous les groupes IAM de votre compte

- Dans le panneau de navigation, sélectionnez Groupes d'utilisateurs.

Pour répertorier les utilisateurs IAM d'un groupe IAM spécifique :

- Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs). Sélectionnez ensuite le nom du groupe pour afficher sa page de détails. Consultez l'onglet Utilisateurs pour voir les membres du groupe.

Pour répertorier tous les groupes IAM auxquels appartient un utilisateur :

- Dans le panneau de navigation, choisissez utilisateurs. Sélectionnez ensuite le nom d'utilisateur pour ouvrir la page des détails de l'utilisateur. Cliquez sur l'onglet Groupes pour voir la liste des groupes auxquels l'utilisateur appartient.

AWS CLI

Pour répertorier tous les groupes IAM de votre compte

- aws iam list-groups

Pour répertorier les utilisateurs d'un groupe IAM spécifique :

- aws iam get-group

Pour répertorier tous les groupes IAM auxquels appartient un utilisateur :

- était un objectif list-groups-for-user

API

Pour répertorier tous les groupes IAM de votre compte

- ListGroups

Pour répertorier les utilisateurs d'un groupe IAM spécifique :

- GetGroup

Pour répertorier tous les groupes IAM auxquels appartient un utilisateur :

- ListGroupsForUser

Modification des utilisateurs dans les groupes IAM

Utilisez les groupes IAM pour appliquer les mêmes politiques d'autorisations sur plusieurs utilisateurs à la fois. Vous pouvez ensuite ajouter des utilisateurs à un groupe IAM ou en supprimer. Cet utile à mesure que des utilisateurs arrivent dans votre organisation et la quittent.

Vérification de l'accès des politiques

Avant de supprimer un groupe, utilisez la page des détails du groupe pour vérifier les membres (utilisateurs IAM) du groupe, les politiques associées au groupe dans l'onglet Autorisations et consulter l'activité récente au niveau du service dans l'onglet Dernière consultation. Ceci permet d'éviter de supprimer involontairement l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Ajouter un utilisateur IAM à un groupe IAM

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), puis choisissez le nom du groupe.
3. Sélectionnez l'onglet Users (Utilisateurs), puis Add users (Ajouter des utilisateurs). Cochez la case à côté de l'utilisateur que vous souhaitez ajouter.
4. Sélectionnez Add users (Ajouter des utilisateurs).

AWS CLI

Exécutez la commande suivante :

- `aws iam add-user-to-group`

API

Appelez l'opération suivante :

- `AddUserToGroup`

Supprimer un utilisateur IAM d'un groupe IAM

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), puis choisissez le nom du groupe.
3. Sélectionnez l'onglet Utilisateurs. Cochez la case en regard de l'utilisateur à supprimer, puis choisissez Remove users (Supprimer des utilisateurs).

AWS CLI

Exécutez la commande suivante :

- `aws iam remove-user-from-group`

API

Appelez l'opération suivante :

- `RemoveUserFromGroup`

Attachement d'une politique à un groupe d'utilisateurs IAM

Vous pouvez associer une politique AWS gérée, c'est-à-dire une politique préécrite fournie par, AWSà un groupe d'utilisateurs, comme expliqué dans les étapes suivantes. Pour attacher une politique gérée par le client, c'est-à-dire une politique pour laquelle vous avez créé des autorisations personnalisées, vous devez d'abord créer la politique. Pour plus d'informations sur la création de politiques gérées par le client, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client.

Pour plus d'informations sur les autorisations et les politiques, consultez Gestion des accès pour AWS resources.

Pour attacher une politique à un groupe IAM

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), puis choisissez le nom du groupe.
3. Sélectionnez l'onglet Autorisations.
4. Choisissez Ajouter des autorisations, puis Attacher des politiques.
5. Les politiques actuellement attachées au groupe d'utilisateurs s'affichent dans la liste Politiques d'autorisation actuelles. Dans la liste des Autres politiques d'autorisation, cochez la case en regard des noms des politiques à attacher. Vous pouvez utiliser la barre de recherche pour filtrer la liste des politiques par type et nom de politique.
6. Sélectionnez la politique que vous souhaitez attacher à votre groupe IAM et choisissez Attacher des politiques.

AWS CLI

Exécutez la commande suivante :

- `aws iam attach-group-policy`

API

Appelez l'opération suivante :

- `AttachGroupPolicy`

Affectation d'un nouveau nom à un groupe d'utilisateurs IAM

Lorsque vous modifiez le nom ou le chemin d'accès d'un groupe d'utilisateurs, voici ce qui suit se produit :

- Toutes les politiques attachées au groupe d'utilisateurs restent dans le groupe sous le nouveau nom.
- Le groupe d'utilisateurs conserve également tous ses utilisateurs sous le nouveau nom.

- L'ID unique du groupe d'utilisateurs demeure le même. Pour plus d'informations sur le caractère unique IDs, consultezIdentifiants uniques.

IAM ne met pas automatiquement à jour les politiques qui font référence au groupe d'utilisateurs en tant que ressource de manière à utiliser le nouveau nom. Par conséquent, vous devez agir avec prudence lorsque vous renommez un groupe d'utilisateurs. Avant de renommer votre groupe d'utilisateurs, vous devez vérifier manuellement toutes vos politiques afin d'identifier les politiques dans lesquelles le nom de ce groupe d'utilisateurs est mentionné. Par exemple, supposons que Bob est responsable des tests pour l'organisation. Bob dispose d'une politique attachée à son entité d'utilisateur IAM lui permettant d'ajouter des utilisateurs au groupe d'utilisateurs Test et d'en supprimer. Si un administrateur renomme le groupe d'utilisateurs (ou modifie le chemin d'accès au groupe), l'administrateur doit également mettre à jour la politique attachée à Bob afin d'utiliser le nouveau nom ou chemin d'accès. Sinon Bob ne sera plus en mesure d'ajouter ou de supprimer des utilisateurs du groupe d'utilisateurs.

Pour rechercher des politiques qui font référence à un groupe IAM en tant que ressource :

1. Dans le panneau de navigation de la console IAM, sélectionnez Policies (Politiques).
2. Triez par la colonne Type pour rechercher vos politiques personnalisées gérées par le client.
3. Sélectionnez le nom de la politique à modifier.
4. Choisissez l'onglet Autorisations, puis Résumé.
5. Choisissez IAM dans la liste des services, si ce service est disponible.
6. Recherchez le nom de votre groupe d'utilisateurs dans la colonne Resource (Ressource).
7. Choisissez Modifier pour changer le nom de votre groupe d'utilisateurs dans la politique.

Pour modifier le nom d'un groupe IAM

Console

1. Dans le panneau de navigation, sélectionnez Groupes d’utilisateurs, puis sélectionnez le nom du groupe.
2. Choisissez Modifier. Tapez le nom du nouveau groupe d'utilisateurs, puis choisissez Save changes (Enregistrer les modifications).

AWS CLI

Exécutez la commande suivante :

- groupe de mise à jour aws iam

API

Appelez l'opération suivante :

- UpdateGroup

Supprimer un groupe IAM

Lorsque vous supprimez un groupe IAM dans la console, celle-ci supprime automatiquement tous les membres du groupe, détache toutes les politiques gérées attachées et supprime toutes les politiques en ligne. Cependant, comme IAM ne supprime pas automatiquement les politiques qui font référence au groupe IAM en tant que ressource, vous devez faire attention lorsque vous supprimez un groupe IAM. Avant de supprimer votre groupe IAM, vérifiez manuellement toutes vos politiques afin d'identifier les politiques dans lesquelles le nom de ce groupe est mentionné. Par exemple, John dispose d'une politique attachée à son entité d'utilisateur IAM lui permettant d'ajouter et supprimer des utilisateurs du groupe d'utilisateurs Test. Si un administrateur supprime le groupe, l'administrateur doit également supprimer la politique attachée à John. Sinon, si l'administrateur recrée le groupe supprimé et lui donne le même nom, les autorisations de John restent en place, même s'il a quitté l'équipe de test.

En revanche, lorsque vous utilisez la CLI, les kits SDK ou l'API pour supprimer un groupe d'utilisateurs, vous devez d'abord supprimer les utilisateurs du groupe. Puis, vous supprimez toutes les politiques en ligne intégrées au groupe IAM. Ensuite, vous détachez toutes les politiques gérées attachées au groupe. Enfin, vous supprimez le groupe IAM lui-même.

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs).

3. Dans la liste des groupes IAM, cochez la case en regard du nom des groupes IAM à supprimer. Vous pouvez utiliser la zone de recherche pour filtrer la liste des groupes IAM par type, autorisations et nom de groupe.
4. Sélectionnez Delete (Supprimer).
5. Dans la zone de confirmation, si vous voulez supprimer un seul groupe, saisissez son nom et choisissez Supprimer. Si vous voulez supprimer plusieurs groupes, saisissez le nombre de groupes IAM à supprimer suivi de **`user groups`** et choisissez Supprimer. Par exemple, si vous souhaitez supprimer trois groupes, saisissez **`3 user groups`**.

AWS CLI

1. Supprimez tous les utilisateurs du groupe IAM.
   - aws iam get-group (pour obtenir la liste des utilisateurs du groupe IAM) et aws iam remove-user-from-group (pour supprimer un utilisateur du groupe IAM)
2. Supprimez toutes les politiques en ligne intégrées au groupe IAM.
   - aws iam list-group-policies (pour obtenir une liste des politiques intégrées du groupe IAM) et aws iam delete-group-policy (pour supprimer les politiques en ligne du groupe IAM)
3. Détachez toutes les politiques gérées attachées au groupe IAM.
   - aws iam list-attached-group-policies (pour obtenir une liste des politiques gérées associées au groupe IAM) et aws iam detach-group-policy (pour détacher une politique gérée du groupe IAM)
4. Supprimez le groupe IAM.
   - aws iam delete-group

API

1. Supprimez tous les utilisateurs du groupe IAM.
   - GetGroup(pour obtenir la liste des utilisateurs du groupe IAM) et RemoveUserFromGroup(pour supprimer un utilisateur du groupe IAM)
2. Supprimez toutes les politiques en ligne intégrées au groupe IAM.

- ListGroupPolicies(pour obtenir une liste des politiques intégrées du groupe IAM) et DeleteGroupPolicy(pour supprimer les politiques en ligne du groupe IAM)

3. Détachez toutes les politiques gérées attachées au groupe IAM.

   - ListAttachedGroupPolicies(pour obtenir une liste des politiques gérées associées au groupe IAM) et DetachGroupPolicy(pour détacher une politique gérée du groupe IAM)

4. Supprimez le groupe IAM.

   - DeleteGroup

Rôles IAM

Un rôle IAM est une identité IAM que vous pouvez créer dans votre compte et qui dispose d'autorisations spécifiques. Un rôle IAM est similaire à un utilisateur IAM, car il s'agit d'une identité AWS avec des politiques d'autorisation qui déterminent ce que l'identité peut et ne peut pas faire dans AWS. En revanche, au lieu d'être associé de manière unique à une personne, un rôle est conçu pour être endossé par tout utilisateur qui en a besoin. En outre, un rôle ne dispose pas d'informations d'identification standard à long terme comme un mot de passe ou des clés d'accès associées. Au lieu de cela, lorsque vous adoptez un rôle, il vous fournit des informations d'identification de sécurité temporaires pour votre session de rôle.

Vous pouvez utiliser des rôles pour déléguer l'accès à des utilisateurs, à des applications ou à des services qui n'ont normalement pas accès à vos AWS ressources. Par exemple, vous souhaiterez peut-être autoriser les utilisateurs de votre AWS compte à accéder à des ressources dont ils ne disposent pas habituellement, ou autoriser les utilisateurs d'un compte à Compte AWS accéder aux ressources d'un autre compte. Vous pouvez également autoriser une application mobile à utiliser AWS des ressources, mais ne pas intégrer de AWS clés dans l'application (où elles peuvent être difficiles à mettre à jour et où les utilisateurs peuvent potentiellement les extraire). Parfois, vous souhaitez donner AWS accès à des utilisateurs dont l'identité est déjà définie en dehors de AWS celle-ci, par exemple dans votre annuaire d'entreprise. Ou, vous pouvez accorder l'accès à votre compte à des tiers afin de leur permettre de réaliser un audit de vos ressources.

Dans ces scénarios, vous pouvez déléguer l'accès aux AWS ressources à l'aide d'un rôle IAM. Cette section présente les rôles et les différentes façons de les utiliser. Elle explique également quand et comment choisir les diverses approches et comment créer, gérer, prendre (endosser) et supprimer des rôles.

???Le gestionnaire d'accès aux comptes est une fonctionnalité IAM qui vous permet d'attribuer de manière centralisée des rôles IAM sur les comptes de votre organisation aux utilisateurs et aux groupes IAM Identity Center. Le gestionnaire d'accès aux comptes permet à votre personnel d'accéder via des rôles IAM avec leur ensemble complet de fonctionnalités. Vous pouvez l'utiliser conjointement avec les ensembles d'autorisations IAM Identity Center ou seul.

> **Note**
>
> Lorsque vous créez le vôtre pour la première fois Compte AWS, aucun rôle n'est créé par défaut. Au fur et à mesure que vous ajoutez des services à votre compte, ils peuvent ajouter des rôles liés à un service pour répondre à leurs cas d'utilisation.
> Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut assumer le rôle d'effectuer une action en votre nom. Service-linked les rôles apparaissent dans votre compte Compte AWS et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.
> Avant de pouvoir supprimer les rôles liés à un service, vous devez d'abord supprimer les ressources qui leur sont associées. Vos ressources sont ainsi protégées, car vous ne pouvez pas involontairement supprimer l'autorisation d'accéder aux ressources.
> Pour plus d'informations sur les services qui prennent en charge l'utilisation de rôles liés à des services, consultez AWS services qui fonctionnent avec IAM et recherchez les services pour lesquels la mention Oui apparaît dans la colonne Service-Linked Rôle. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Rubriques

- Quand créer un utilisateur IAM (au lieu d'un rôle)
- Termes et concepts relatifs aux rôles
- Ressources supplémentaires
- Le problème de l'adjoint confus
- Scénarios courants de rôles IAM
- Création d'un rôle IAM
- Gestion des rôles IAM
- Méthodes pour assumer un rôle

Quand créer un utilisateur IAM (au lieu d'un rôle)

Nous vous recommandons de n'utiliser les utilisateurs IAM que pour les cas d'utilisation non pris en charge par la fédération d'identité. Voici certaines des fonctionnalités les plus utilisées :

- Charges de travail qui ne peuvent pas utiliser de rôles IAM — Vous pouvez exécuter une charge de travail à partir d'un emplacement qui a besoin d'accéder à AWS. Dans certains cas, vous ne pouvez pas utiliser les rôles IAM pour fournir des informations d'identification temporaires, par exemple pour les WordPress plugins. Dans ces situations, utilisez les clés d'accès à long terme de l'utilisateur IAM pour cette charge de travail afin de vous authentifier auprès de AWS.
- Third-party AWS clients : si vous utilisez des outils qui ne prennent pas en charge l'accès à IAM Identity Center, tels que des AWS clients tiers ou des fournisseurs qui ne sont pas hébergés sur AWS, utilisez des clés d'accès utilisateur à long terme IAM.
- AWS CodeCommit accès : si vous l'utilisez CodeCommit pour stocker votre code, vous pouvez utiliser un utilisateur IAM avec des clés SSH ou des informations d'identification spécifiques au service pour vous authentifier CodeCommit auprès de vos référentiels. Nous vous recommandons de procéder ainsi en plus de vous servir d'un utilisateur dans IAM Identity Center pour une authentification ordinaire. Les utilisateurs d'IAM Identity Center sont les membres de votre personnel qui ont besoin d'accéder à vos applications Comptes AWS ou à vos applications cloud. Pour permettre aux utilisateurs d'accéder à vos CodeCommit référentiels sans configurer les utilisateurs IAM, vous pouvez configurer l'git-remote-codecommitutilitaire. Pour plus d'informations sur IAM et CodeCommit, consultezInformations d'identification IAM pour CodeCommit : informations d'identification Git, clés SSH et AWS clés d'accès. Pour plus d'informations sur la configuration de l'git-remote-codecommitutilitaire, consultez la section Connexion à AWS CodeCommit des référentiels avec des informations d'identification rotatives dans le Guide de AWS CodeCommit l'utilisateur.
- Amazon Keyspaces (for Apache Cassandra) access (Accès Amazon Keyspaces (pour Apache Cassandra)) : dans une situation où vous n'êtes pas en mesure de vous servir des utilisateurs dans IAM Identity Center, par exemple à des fins de test de compatibilité avec Cassandra, vous pouvez utiliser un utilisateur IAM avec des informations d'identification spécifiques au service pour vous authentifier auprès d'Amazon Keyspaces. Les utilisateurs d'IAM Identity Center sont les membres de votre personnel qui ont besoin d'accéder à vos applications Comptes AWS ou à vos applications cloud. Vous pouvez également vous connecter à Amazon Keyspaces à l'aide d'informations d'identification temporaires. Pour de plus d'informations, veuillez consulter Utilisation d'informations d'identification temporaires pour se connecter à Amazon Keyspaces à l'aide d'un

rôle IAM et du plugin Sigv4 dans le Guide du développeur Amazon Keyspaces (pour Apache Cassandra).

- Accès d’urgence : dans une situation où vous n’avez pas accès à votre fournisseur d’identité et où vous devez prendre des mesures sur votre Compte AWS. La création d'utilisateurs IAM bénéficiant d'un accès d'urgence peut faire partie de votre plan de résilience. Nous vous recommandons de veiller à ce que les informations d'identification des utilisateurs d'urgence soient étroitement contrôlées et sécurisées à l'aide de l'authentification multifactorielle (MFA).

Termes et concepts relatifs aux rôles

Voici quelques termes de base pour vous aider à vous familiariser avec les rôles.

Rôle

Une identité IAM que vous pouvez créer dans votre compte et qui dispose d'autorisations spécifiques. Un rôle IAM présente des similitudes avec un utilisateur IAM. Les rôles et les utilisateurs sont tous deux des identités AWS avec des politiques d'autorisations qui déterminent ce que l'identité peut ou ne peut pas faire dans AWS. En revanche, au lieu d'être associé de manière unique à une personne, un rôle est conçu pour être endossé par tout utilisateur qui en a besoin. En outre, un rôle ne dispose pas d’informations d’identification standard à long terme comme un mot de passe ou des clés d’accès associées. Au lieu de cela, lorsque vous adoptez un rôle, il vous fournit des informations d’identification de sécurité temporaires pour votre session de rôle.

Les rôles peuvent être endossés par :

- Un utilisateur IAM dans le même Compte AWS ou un autre Compte AWS
- Rôles IAM dans le même compte
- Principes de service, à utiliser avec des AWS services et fonctionnalités tels que :
  - Services qui vous permettent d’exécuter du code sur des services de calcul, tels qu’Amazon EC2 ou AWS Lambda
  - Fonctionnalités qui exécutent des actions sur vos ressources en votre nom, comme la réplication d’objets Amazon S3
  - Services qui fournissent des informations d'identification de sécurité temporaires à vos applications qui s'exécutent en dehors de AWS, tels que IAM Roles Anywhere ou Amazon ECS Anywhere

- Un utilisateur externe authentifié par un service de fournisseur d'identité (IdP) externe, compatible avec SAML 2.0 ou OpenID Connect.

AWS rôle de service

Un rôle de service est un rôle IAM qu'un service endosse pour accomplir des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer un rôle de service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.

AWS rôle lié à un service

Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut assumer le rôle d'effectuer une action en votre nom. Service-linked les rôles apparaissent dans votre compte Compte AWS et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

> **Note**
>
> Si vous utilisez déjà un service lorsqu'il commence à prendre en charge des rôles liés à un service, vous pouvez recevoir un e-mail vous informant de l'existence d'un nouveau rôle sur votre compte. Dans ce cas, le service crée automatiquement le rôle lié à un service sur votre compte. Aucune action de votre part n'est requise pour prendre ce rôle en charge et il est préférable de ne pas le supprimer manuellement. Pour de plus amples informations, veuillez consulter Un nouveau rôle est apparu dans mon AWS compte.

Pour plus d'informations sur les services qui prennent en charge l'utilisation de rôles liés à des services, consultez AWS services qui fonctionnent avec IAM et recherchez les services pour lesquels la mention Oui apparaît dans la colonne Service-Linked Rôle. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service. Pour de plus amples informations, veuillez consulter Créer un rôle lié à un service.

Création de chaînes de rôles

La création de chaînes de rôles se produit lorsque vous utilisez un rôle pour assumer un second rôle. Vous pouvez effectuer le chaînage des rôles via le Console de gestion AWS en changeant de rôle, d'API ou d'API. AWS CLI Par exemple, `RoleA` dispose de l'autorisation d'assumer `RoleB`. Vous pouvez autoriser User1 à assumer `RoleA` en utilisant ses informations d'identification utilisateur à long terme dans le cadre du fonctionnement de l' AssumeRoleAPI. Cette opération

renvoie les informations d'identification à court terme de `RoleA`. Avec le chaînage de rôles, vous pouvez utiliser les informations d'identification à court terme de `RoleA` pour permettre à User1 d'assumer `RoleB`.

Lorsque vous endossez un rôle, vous pouvez passer une balise de session et la définir comme transitive. Les balises de session transitives sont transmises à toutes les sessions suivantes d'une chaîne de rôles. Pour en savoir plus sur les balises de session, veuillez consulter Transmettre les balises de session AWS STS.

Le chaînage des rôles limite votre Console de gestion AWS session AWS CLI ou celle de votre session de rôle d' AWS API à une heure maximum. Elle s'applique quelle que soit la durée maximale de session configurée pour les rôles individuels. Lorsque vous utilisez l'opération AssumeRole API pour assumer un rôle, vous pouvez spécifier la durée de votre session de rôle à l'aide du `DurationSeconds` paramètre. Vous pouvez spécifier une valeur de paramètre jusqu'à 43200 secondes (12 heures), en fonction du paramètre de durée de session maximale de votre rôle. Toutefois, si vous endossez un rôle à l'aide de la création de chaînes de rôles et que vous définissez une valeur de paramètre `DurationSeconds` supérieure à une heure, l'opération échoue.

Pour plus d'informations sur le passage à un rôle dans le Console de gestion AWS, consultezBasculer d'un utilisateur à un rôle IAM (console).

Délégation

L'octroi à une personne d'autorisations lui permettant d'accéder aux ressources que vous contrôlez. La délégation implique la mise en place d'une confiance entre deux comptes. Le premier est le compte propriétaire de la ressource (le compte de confiance). Le second est le compte qui contient les utilisateurs qui doivent accéder à la ressource (le compte approuvé). Les comptes d'approbation et approuvés peuvent être :

- Un même compte.
- Comptes distincts se trouvant tous deux sous le contrôle de votre organisation.
- Deux comptes appartenant à des organisations différentes.

Pour déléguer l'autorisation d'accès à une ressource, vous créez un rôle IAM dans le compte d'approbation auquel sont attachées deux politiques. La politique d'autorisation accorde à l'utilisateur du rôle les autorisations nécessaires pour exécuter les tâches prévues sur la ressource. La politique d'approbation détermine les membres du compte autorisés à endosser le rôle.

Lorsque vous créez une politique de confiance, vous ne pouvez pas spécifier de caractère générique (*) dans le cadre d'un ARN dans l'élément principal. La politique d'approbation est attachée au rôle dans le compte d'approbation, et représente la moitié des autorisations. L'autre moitié est une politique d'autorisation attachée à l'utilisateur dans le compte approuvé qui autorise cet utilisateur à prendre ou endosser le rôle. Un utilisateur qui endosse un rôle temporairement abandonne ses propres autorisations, de manière à adopter celles du rôle. Lorsque l'utilisateur quitte le rôle ou cesse de l'utiliser, ses autorisations d'origine sont restaurées. Un paramètre supplémentaire appelé ID externe permet de sécuriser l'utilisation de rôles entre des comptes qui ne sont pas contrôlés par la même organisation.

Politique d'approbation

Document de politique JSON dans lequel vous définissez les principaux en lesquels vous avez confiance pour endosser le rôle. Une politique d'approbation de rôle est une politique basée sur les ressources requise qui est attachée à un rôle dans IAM. Les principaux que vous pouvez spécifier dans la politique d'approbation comprennent les utilisateurs, les rôles, les comptes et les services. Pour plus d'informations, consultez Comment utiliser les politiques de confiance avec les rôles IAM dans le Blog de sécuritéAWS .

Rôle pour l'accès entre comptes

Un rôle qui octroie l'accès aux ressources d'un compte à un principal approuvé d'un autre compte. Les rôles constituent le principal moyen d'accorder l'accès intercompte. Cependant, certains AWS services vous permettent d'associer une politique directement à une ressource (au lieu d'utiliser un rôle en tant que proxy). Ces politiques sont appelées politiques basées sur les ressources et vous pouvez les utiliser pour accorder aux principaux titulaires d'un autre Compte AWS accès à la ressource. Ces ressources incluent notamment les compartiments Amazon Simple Storage Service (S3), les coffres Amazon Glacier, les rubriques Amazon Simple Notification Service (SNS) et les files d'attente Amazon Simple Queue Service (SQS). Pour connaître les services qui prennent en charge les politiques basées sur les ressources, veuillez consulter AWS services qui fonctionnent avec IAM. Pour plus d'informations sur les politiques basées sur les ressources, consultez Accès intercompte aux ressources dans IAM.

Ressources supplémentaires

Les ressources suivantes peuvent vous aider à en savoir plus sur la terminologie IAM relative aux rôles IAM.

- Les principaux sont des entités AWS qui peuvent effectuer des actions et accéder à des ressources. Un principal peut être un Utilisateur racine d'un compte AWS utilisateur IAM ou un rôle. Un principal qui représente l'identité d'un AWS service est un principal de service. Utilisez l'élément Principal dans les politiques d'approbation des rôles pour définir les principaux auxquels vous faites confiance pour assumer le rôle.

  Pour plus d'informations et des exemples de principaux que vous pouvez autoriser à assumer un rôle, consultez AWS Éléments de politique JSON : Principal.

- La fédération d'identité crée une relation de confiance entre un fournisseur d'identité externe et AWS. Vous pouvez utiliser votre fournisseur OpenID Connect (OIDC) ou Security Assertion Markup Language (SAML) 2.0 pour gérer les utilisateurs autorisés à accéder aux ressources AWS . Lorsque vous utilisez OIDC et SAML 2.0 pour configurer une relation de confiance entre ces fournisseurs d'identité externes et AWS que l'utilisateur se voit attribuer un rôle IAM. L'utilisateur reçoit également des informations d'identification temporaires qui lui permettent d'accéder à vos ressources AWS .

  Pour plus d'informations sur les principaux fédérés, consultez Fournisseurs d'identité et fédération au sein AWS.

- Les principaux fédérés sont des identités existantes provenant de l' Directory Service annuaire des utilisateurs de votre entreprise ou d'un fournisseur OIDC. AWS attribue un rôle à un principal fédéré lorsque l'accès est demandé via un fournisseur d'identité.

  Pour plus d'informations sur les principaux fédérés SAML ou OIDC, consultez sessions d'utilisateur fédéré et rôle.

- Les politiques d'autorisation sont des politiques basées sur l'identité qui définissent les actions et les ressources que le rôle peut utiliser. Le document est écrit conformément aux règles du langage de politique IAM.

  Pour de plus amples informations, veuillez consulter Référence de politique JSON IAM.

- Les limites des autorisations sont une fonctionnalité avancée dans laquelle vous utilisez des politiques pour limiter les autorisations maximales qu'une politique basée sur l'identité peut accorder à un rôle. Vous ne pouvez pas appliquer une limite d'autorisations à un rôle lié à un service.

  Pour de plus amples informations, veuillez consulter Limites d'autorisations pour les entités IAM.

Le problème de l'adjoint confus

Le problème de député confus est un problème de sécurité dans lequel une entité qui n’est pas autorisée à effectuer une action peut contraindre une entité plus privilégiée à le faire. Pour éviter cela, AWS fournit des outils qui vous aident à protéger votre compte si vous fournissez à des tiers (comptes croisés) ou à d'autres AWS services (appelés interservices) un accès aux ressources de votre compte.

Il peut arriver que vous deviez accorder à un tiers l'accès à vos AWS ressources (accès délégué). Par exemple, vous décidez de faire appel à une société tierce appelée Example Corp pour surveiller vos coûts Compte AWS et vous aider à optimiser les coûts. Afin de suivre vos dépenses quotidiennes, Example Corp doit accéder à vos AWS ressources. Example Corp surveille également de nombreux autres Comptes AWS pour d'autres clients. Vous pouvez utiliser un rôle IAM pour établir une relation de confiance entre votre compte Compte AWS et le compte Example Corp. Un aspect important de ce scénario est l’ID externe, un identifiant facultatif que vous pouvez utiliser dans une politique d’approbation des rôles IAM afin de désigner l’utilisateur autorisé à endosser le rôle. La fonction principale de l’ID externe est de traiter et de prévenir le problème du délégué confus.

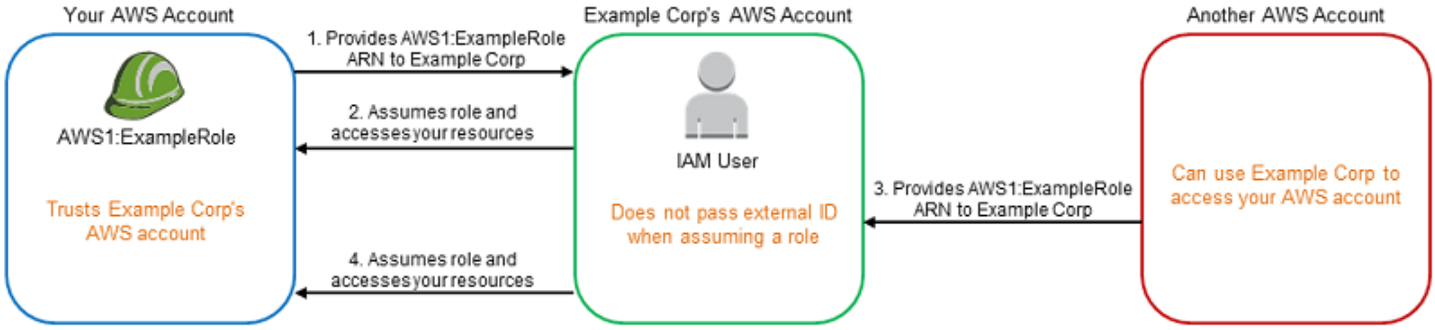
Certains AWS services (services d'appel) utilisent leur principal de AWS service pour accéder aux AWS ressources d'autres AWS services (appelés services). Dans certaines de ces interactions de service, vous pouvez configurer les services d'appel pour communiquer avec les ressources d'un service appelé d'un autre Compte AWS. C'est le cas, par exemple, AWS CloudTrail de la configuration pour écrire dans un compartiment Amazon S3 central situé dans un autre compartiment Compte AWS. Le service d'appel est autorisé à accéder à votre compartiment S3 conformément à la politique du compartiment S3 en ajoutant une instruction d'autorisation pour`cloudtrail.amazonaws.com`. CloudTrail

Lorsqu'un principal de AWS service d'un service appelant accède à une ressource à partir d'un service appelé, la politique de ressources du service appelé autorise uniquement le principal de AWS service, et non l'acteur qui a configuré le service d'appel. Par exemple, un compartiment S3 qui fait confiance au principal du CloudTrail service sans aucune condition peut recevoir CloudTrail des journaux de la Comptes AWS part d'un administrateur de confiance configuré, mais également CloudTrail des journaux d'un acteur non autorisé Compte AWS, s'il connaît le nom du compartiment S3.

Le problème des adjoints confus se pose lorsqu'un acteur utilise la confiance du directeur de AWS service d'un service pour accéder à des ressources auxquelles il n'est pas censé avoir accès.

Cross-account prévention confuse des adjoints

Le diagramme suivant illustre le problème de l'adjoint confus entre comptes.



Ce scénario suppose que :

- AWS 1 est votre Compte AWS.
- AWS 1 : ExampleRole est un rôle dans votre compte. La politique d'approbation du rôle approuve Example Corp en désignant le compte AWS d'Example Corp comme celui qui peut endosser le rôle.

Voici ce qui se produit :

1. Lorsque vous commencez à utiliser le service d'Example Corp, vous fournissez l'ARN AWS 1 : ExampleRole à Example Corp.
2. Example Corp utilise cet ARN de rôle pour obtenir des informations d'identification de sécurité temporaires afin d'accéder aux ressources de votre Compte AWS. Ainsi, vous approuvez Example Corp en tant que « député » autorisé à agir pour votre compte.
3. Un autre AWS client commence également à utiliser le service d'Example Corp, et ce client fournit également l'ARN AWS 1 : ExampleRole for Example Corp à utiliser. On peut supposer que l'autre client a appris ou deviné le AWS 1 : ExampleRole, ce qui n'est pas un secret.
4. Lorsque l'autre client demande à Example Corp d'accéder aux AWS ressources (qu'il prétend être) de son compte, Example Corp utilise AWS 1 : ExampleRole pour accéder aux ressources de votre compte.

C'est ainsi que l'autre client peut obtenir un accès non autorisé à vos ressources. Du fait que cet autre client a été en mesure de tromper Example Corp pour agir involontairement sur vos ressources, Example Corp est à présent un « député confus ».

Exemple Corp peut résoudre le problème de l'adjoint confus en exigeant que vous incluiez la vérification de la condition `ExternalId` dans la politique d'approbation du rôle. Exemple Corp génère une valeur `ExternalId` unique pour chaque client et utilise cette valeur dans sa demande de prise en charge du rôle. La valeur `ExternalId` doit être unique parmi les clients d'Example Corp et contrôlée par Example Corp, et non par ses clients. C'est pourquoi vous l'obtenez d'Example Corp et que vous ne créez pas le vôtre. Cela évite à Example Corp d'être un adjoint confus et d'accorder l'accès aux AWS ressources d'un autre compte.

Dans notre scénario, imaginez que l'identifiant unique d'Example Corp pour vous soit 12345, et que son identifiant pour l'autre client soit 67890. Ces identifiants sont simplifiés pour ce scénario. En général, ces identifiants sont des GUID. Supposons que ces identifiants sont uniques pour chaque client d'Example Corp, ils constituent des valeurs sensibles à utiliser pour l'ID externe.

Example Corp vous attribue la valeur d'ID externe 12345. Vous devez ensuite ajouter un élément `Condition` à la politique d'approbation du rôle qui nécessite que la valeur de `sts:ExternalId` soit 12345, comme suit :

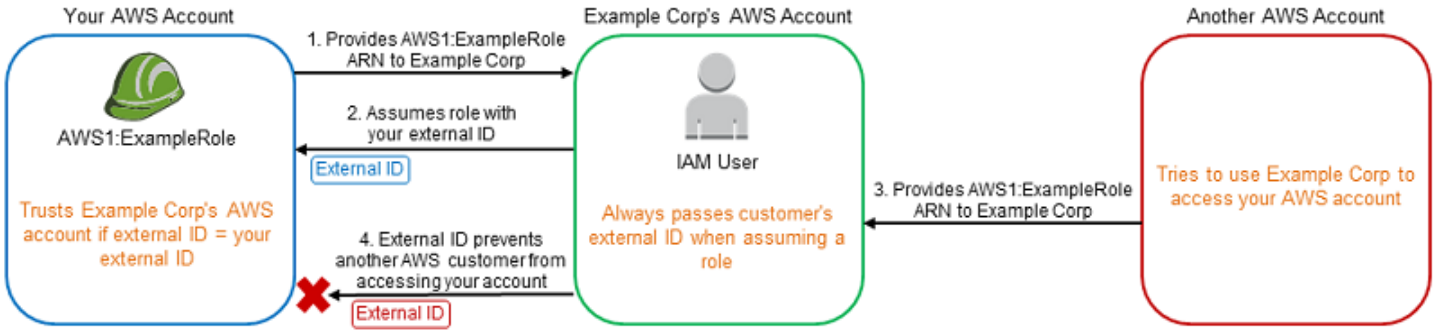
JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "AWS": "Example Corp's AWS Account ID"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "sts:ExternalId": "12345"
      }
    }
  }
}
```

L'élément Condition de cette politique permet à Example Corp d'assumer le rôle uniquement lorsque l'appel d' AssumeRole API inclut la valeur d'ID externe 12345. Example Corp s'assure que chaque fois qu'elle assume un rôle au nom d'un client, elle inclut toujours la valeur d'identifiant externe de ce client dans l' AssumeRole appel. Même si un autre client fournit votre ARN à Example Corp, il ne

peut pas contrôler l'ID externe qu'Example Corp inclut dans sa demande AWS. Cela permet d'éviter qu'un client non autorisé obtienne l'accès à vos ressources.

Le schéma suivant illustre ce processus.

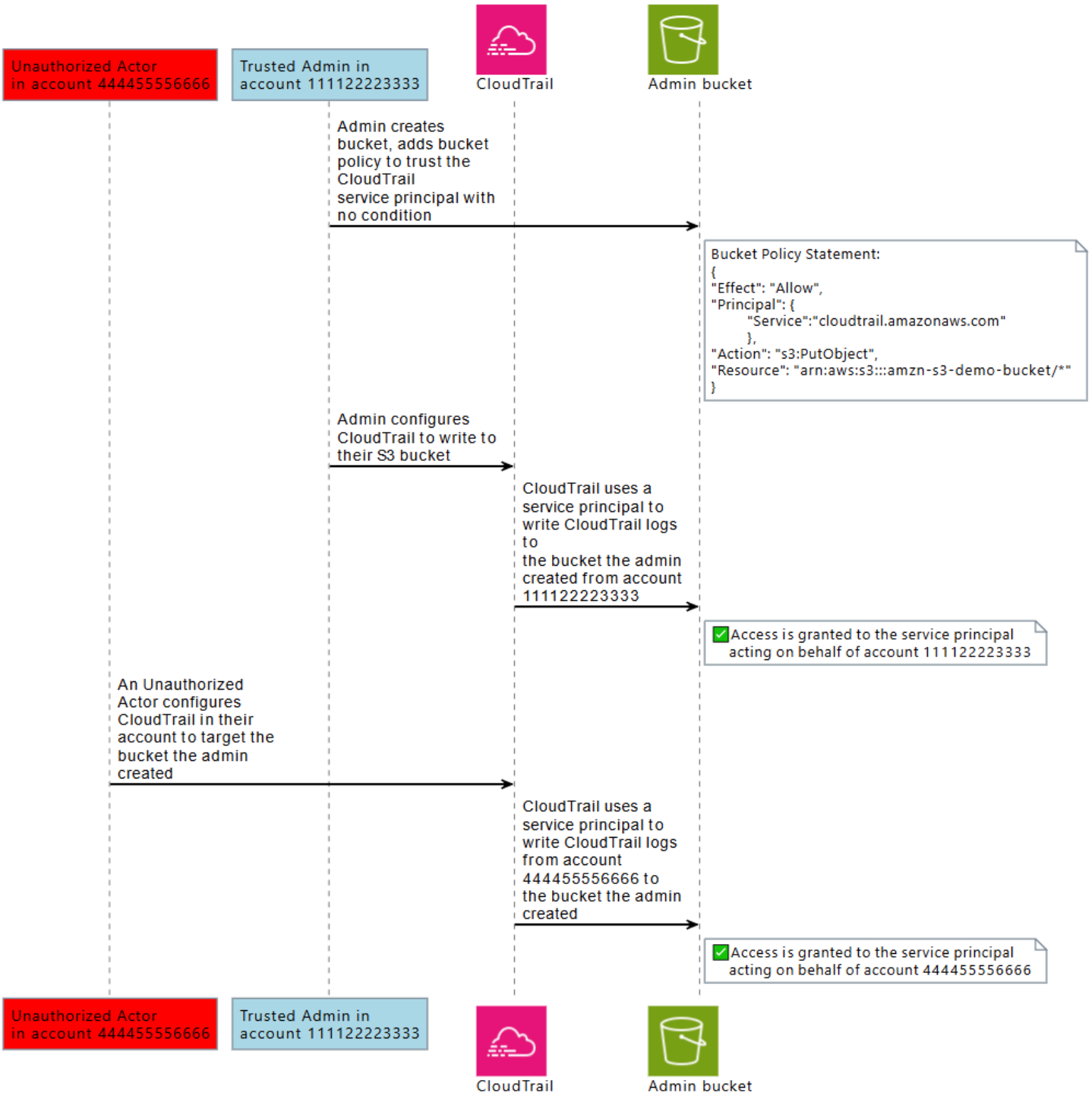


1. Comme précédemment, lorsque vous commencez à utiliser le service d'Example Corp, vous fournissez l'ARN AWS 1 : ExampleRole à Example Corp.
2. Lorsque Example Corp utilise ce rôle ARN pour assumer le rôle AWS 1 : ExampleRole, Example Corp inclut votre ID externe (12345) dans l'appel d' AssumeRole API. L'ID externe correspond à la politique de confiance du rôle. Ainsi, l'appel d' AssumeRole API aboutit et Example Corp obtient des informations de sécurité temporaires pour accéder aux ressources de votre. Compte AWS
3. Un autre AWS client commence également à utiliser le service d'Example Corp et, comme auparavant, ce client fournit également l'ARN AWS 1 : ExampleRole for Example Corp à utiliser.
4. Mais cette fois, lorsque Example Corp tente d'assumer le rôle AWS 1 : ExampleRole, elle fournit l'ID externe associé à l'autre client (67890). L'autre client est dans l'impossibilité de changer cela. Example Corp procède ainsi, car la demande pour utiliser le rôle provient de l'autre client, 67890 indique donc les circonstances dans lesquelles Example Corp agit. Comme vous avez ajouté une condition avec votre propre identifiant externe (12345) à la politique de confiance de AWS 1 : ExampleRole, l'appel d' AssumeRole API échoue. L'autre client se voit refuser l'autorisation d'accéder aux ressources de votre compte (indiqué par la croix « X » rouge dans le diagramme).

L'ID externe empêche tout autre client de forcer Example Corp de manière trompeuse à accéder involontairement à vos ressources.

Cross-service prévention confuse des adjoints

Le schéma suivant illustre le problème de confusion entre les services adjoints en utilisant l' CloudTrail exemple d'interaction avec Amazon S3, dans lequel un acteur non autorisé écrit CloudTrail des journaux dans un compartiment Amazon S3 auquel il n'est pas autorisé à accéder.



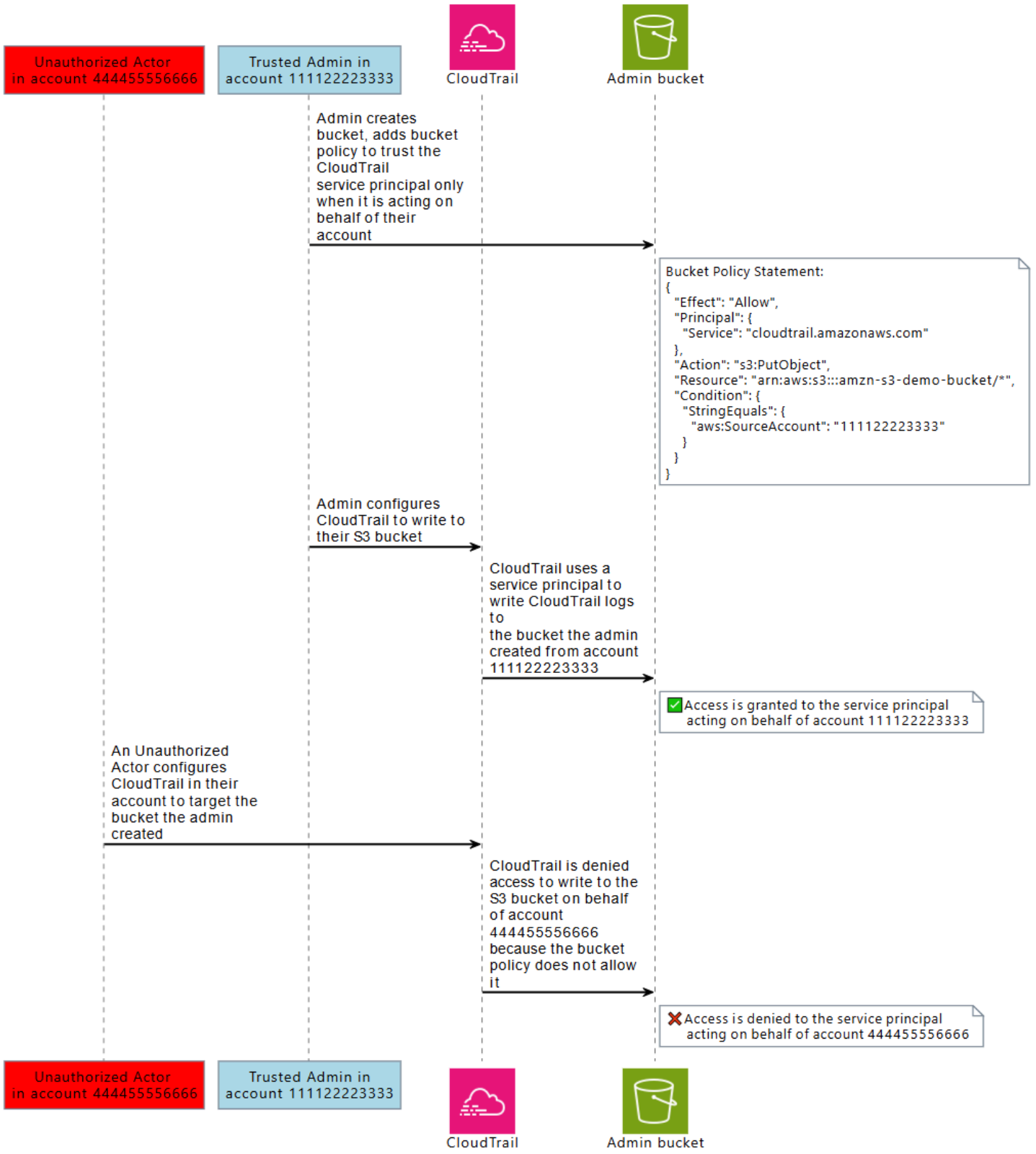
Pour éviter qu'un acteur non autorisé n'utilise la confiance d'un AWS mandant pour accéder à vos ressources, les responsables de AWS service incluent des informations sur la AWS ressource et l' AWS organisation pour laquelle ils agissent pour le compte de ceux-ci. Compte AWS

Ces informations sont disponibles sous forme de valeurs de clé de condition globale qui peuvent être utilisées dans une politique de ressources ou dans une politique de contrôle des ressources pour les

demandes effectuées par les principaux AWS fournisseurs de service. Nous vous recommandons d'utiliser aws : SourceArnaws : SourceAccount,aws : SourceOrg ID, ou aws : SourceOrgPaths dans vos politiques de ressources chaque fois qu'un directeur de AWS service est autorisé à accéder à l'une de vos ressources. Ces clés de condition vous permettent de tester dans vos politiques de ressources, ou vos politiques de contrôle des ressources, que les responsables de AWS service accédant à vos ressources le font au nom des AWS ressources Comptes AWS, ou comme AWS Organizations vous vous y attendez.

- `aws:SourceArn`À utiliser pour autoriser un directeur de AWS service à accéder à vos ressources au nom d'une ressource spécifique, telle qu'un AWS CloudTrail sentier ou une AppStream flotte spécifique.
- `aws:SourceAccount`À utiliser pour autoriser un directeur AWS de service à accéder à vos ressources pour le compte d'une personne en particulier Compte AWS.
- `aws:SourceOrgID`À utiliser pour autoriser un directeur AWS de service à accéder à vos ressources pour le compte d'un particulier AWS Organizations.
- `aws:SourceOrgPaths`À utiliser pour autoriser le principal de AWS service à accéder à vos ressources au nom d'un AWS Organizations chemin spécifique.

Le schéma suivant illustre le scénario d'un adjoint confus entre services lorsqu'une ressource est configurée avec la clé de contexte de condition `aws:SourceAccount` globale et qu'un acteur non autorisé d'un autre compte tente d'accéder à AWS des ressources auxquelles il n'est pas censé avoir accès.



L'utilisation des clés de condition globales `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` et `aws:SourceOrgPaths` dans une politique vous aide à vous assurer que les

principaux de service accèdent à vos ressources en votre nom. Nous recommandons d'utiliser ces clés de condition chaque fois que l'accès à l'une de vos ressources est accordé à un principal AWS de service.

> **Note**
>
> Certaines Service AWS interactions comportent des contrôles supplémentaires qui permettent de se protéger contre les problèmes liés à la confusion entre les services qui mettent à l'épreuve l'accès des utilisateurs à une ressource. Par exemple, lorsqu'une attribution de clé KMS est accordée à un Service AWS, AWS KMS utilise le contexte de chiffrement associé à la ressource et l'attribution de clé pour éviter les problèmes de confusion entre les services auxiliaires.
> Veuillez consulter la documentation des services que vous utilisez pour plus d'informations sur les mécanismes spécifiques à chaque service qui peuvent aider à éviter les risques d'adjoint confus entre les services, et pour savoir si `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID`, et `aws:SourceOrgPaths` sont pris en charge.

Cross-service a confondu la protection des adjoints avec des politiques basées sur les ressources

L'exemple de politique suivant accorde au principal de service l'`cloudtrail.amazonaws.com`accès au compartiment Amazon S3, arn:aws:s3 : ::amzn-s3-demo-bucket1, uniquement lorsque le principal de service agit pour le compte de 111122223333. Compte AWS

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CloudTrailAclCheck",
            "Effect": "Allow",
            "Principal": {"Service": "cloudtrail.amazonaws.com"},
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1",
            "Condition": {
                "StringEquals": {
```

```
                    "aws:SourceAccount": "111122223333"
                }
            }
        },
        {
            "Sid": "AWSCloudTrailWrite",
            "Effect": "Allow",
            "Principal": {"Service": "cloudtrail.amazonaws.com"},
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/[optionalPrefix]/Logs/
myAccountID/*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                }
            }
        }
    ]
}
```

Cet exemple de politique de compartiment accorde au principal du service l'`appstream.amazonaws.com`accès au script PowerShell examplefile.psh sur le site s3://amzn-s3-demo-bucket2 uniquement lorsqu'il agit pour le compte de la AppStream flotte Amazon spécifiée, en spécifiant l'ARN de la flotte avec. `aws:SourceArn`

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "appstream.amazonaws.com"
                ]
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket2/examplefile.psh",
            "Condition": {
                "ArnEquals": {
```

```
                    "aws:SourceArn": "arn:aws:appstream:us-
east-1:111122223333:fleet/ExampleFleetName"
                }
            }
        }
    ]
}
```

Cross-service a confondu la protection des adjoints avec les politiques de contrôle des ressources

Vous pouvez utiliser les politiques de contrôle des ressources (RCP) pour appliquer des contrôles d'adjoint confus entre les services aux ressources des Services AWS pris en charge. Les RCP vous permettent d'appliquer de manière centralisée des contrôles d'adjoint confus entre les services sur vos ressources. Vous pouvez utiliser des clés de condition comme `aws:SourceOrgId` et `aws:SourceOrgPaths` avec des RCP attachés à vos AWS Organizations unités organisationnelles (UO) ou Comptes AWS au sein de votre organisation sans ajouter d'instructions à des politiques spécifiques basées sur les ressources. Pour plus d'informations sur les RCP et les services pris en charge, consultez Politiques de contrôle des ressources (RCP) dans le Guide de l'utilisateur AWS Organizations .

Dans l'exemple suivant, le RCP refuse aux principaux de AWS service l'accès aux compartiments Amazon S3 de vos comptes membres lorsque la valeur n'`aws:SourceOrgID`est pas égale à o-. ExampleOrg Une autorisation correspondante doit être présente dans la politique basée sur les ressources du compartiment S3 pour autoriser Service AWS les principaux ayant une `SourceOrgID` valeur égale à o-. ExampleOrg

Cette politique n'applique le contrôle que sur les demandes des principaux de service (`"Bool": {"aws:PrincipalIsAWSService": "true"}`) qui ont la clé `aws:SourceAccount` présente (`"Null": {"aws:SourceAccount": "false"}`), de sorte que les intégrations de services qui ne nécessitent pas l'utilisation de la clé de condition et les appels de vos principaux ne sont pas impactés. Si la clé de condition `aws:SourceAccount` est présente dans le contexte de la demande, la condition Null sera évaluée comme vraie, ce qui entraînera l'application de `aws:SourceOrgID`. Nous utilisons `aws:SourceAccount` au lieu de `aws:SourceOrgID` dans l'opérateur de condition Null afin que le contrôle s'applique toujours si la demande provient d'un compte qui n'appartient pas à une organisation.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RCPEnforceConfusedDeputyProtectionForS3",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceOrgID": "o-ExampleOrg"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
        }
      }
    }
  ]
}
```

Scénarios courants de rôles IAM

Comme pour la plupart des AWS fonctionnalités, vous pouvez généralement utiliser un rôle de deux manières : de manière interactive dans la console IAM ou par programmation avec les outils pour Windows PowerShell ou l' AWS CLI API.

- Les utilisateurs IAM de votre compte qui utilisent la console IAM peuvent basculer vers un rôle leur permettant d'utiliser temporairement les autorisation du rôle dans la console. Les utilisateurs abandonnent leurs autorisations d'origine et acceptent les autorisations attribuées au rôle. Lorsque les utilisateurs quittent le rôle, leurs autorisations d'origine sont restaurées.
- Une application ou un service proposé par AWS (comme Amazon EC2) peut assumer un rôle en demandant des informations d'identification de sécurité temporaires pour un rôle auquel envoyer

des demandes programmatiques. AWS Vous utilisez ce rôle ainsi pour éviter de partager ou de conserver des informations d'identification de sécurité à long terme (par exemple, en créant un utilisateur IAM) pour chaque entité qui doit accéder à une ressource.

> **Note**
>
> Ce manuel utilise les phrases passer à un rôle et endosser un rôle de manière interchangeable.

La forme d'utilisation des rôles la plus simple consiste à accorder à vos utilisateurs IAM l'autorisation de basculer vers des rôles que vous créez au sein de votre propre entreprise ou d'une autre Compte AWS. Ils peuvent changer de rôles facilement à l'aide de la console IAM pour utiliser des autorisations dont vous ne souhaitez pas qu'ils disposent d'ordinaire et il leur suffit de quitter le rôle pour renoncer à ces autorisations. Cela permet d'empêcher un accès accidentel à des ressources sensibles ou leur modification involontaire.

Pour utiliser les rôles de manière plus complexe, comme accorder l'accès à des applications et des services, ou à des utilisateurs externes fédérés, vous pouvez appeler l'API `AssumeRole`. Cet appel d'API renvoie un ensemble d'informations d'identification temporaires que l'application peut utiliser dans les appels d'API suivants. Les tentatives d'actions avec les informations d'identification temporaires ne disposent que des autorisations accordées par le rôle associé. Une application n'a pas besoin de « quitter » le rôle de la même manière qu'un utilisateur dans la console. L'application arrête simplement d'utiliser les informations d'identification temporaires et reprend ses appels avec les informations d'identification d'origine.

Les utilisateurs fédérés se connectent à l'aide des informations d'identification d'un fournisseur d'identité (IdP). AWS fournit ensuite des informations d'identification temporaires à l'IdP de confiance à transmettre à l'utilisateur pour les inclure dans les demandes de AWS ressources ultérieures. Ces informations d'identification fournissent des autorisations accordées au rôle attribué.

Cette section présente les scénarios suivants :

- Donnez accès à un utilisateur IAM dans un compte Compte AWS que vous possédez pour accéder aux ressources d'un autre compte que vous possédez
- Fournir un accès aux charges de travail qui n'appartiennent pas à AWS
- Octroyer un accès aux utilisateurs IAM dans des Comptes AWS appartenant à des tierces parties

- Fournir un accès aux services offerts par AWS deux AWS ressources
- Octroi de l'accès à des utilisateurs authentifiés en externe (fédération d'identité)

Accès d'un utilisateur IAM à un autre Compte AWS que tu possèdes

Vous pouvez autoriser vos utilisateurs IAM à passer à des rôles au sein du vôtre Compte AWS ou à des rôles définis dans d'autres rôles Comptes AWS que vous possédez.

> **Note**
>
> Si vous souhaitez accorder l'accès à un compte que ne vous appartenant pas ou que vous ne contrôlez pas, consultez Accès à Comptes AWS appartenant à des tiers ultérieurement dans cette rubrique.

Imaginons que vous disposez d'instances Amazon EC2 qui sont critiques pour votre organisation. Au lieu d'accorder directement à vos utilisateurs l'autorisation de supprimer les instances, vous pouvez créer un rôle avec ces privilèges. Ensuite, autorisez les administrateurs à passer au rôle lorsqu'ils ont besoin de supprimer une instance. Cela ajoute les couches de protection suivantes aux instances :

- Vous devez accorder explicitement à vos utilisateurs l'autorisation d'endosser le rôle.
- Vos utilisateurs doivent passer activement au rôle à l'aide de l'API Console de gestion AWS ou assumer le rôle à l'aide de l' AWS API AWS CLI ou.
- Vous pouvez ajouter une protection d'authentification multi-facteur (MFA) au rôle afin que seuls les utilisateurs qui se connectent avec un dispositif MFA puissent endosser le rôle. Pour apprendre à configurer un rôle afin que les utilisateurs qui l'endossent doivent d'abord être authentifiés à l'aide de l'authentification multi-facteur (MFA), consultez Accès sécurisé aux API avec MFA.

Nous vous recommandons d'utiliser cette approche pour appliquer le principe du moindre privilège. Cela implique de restreindre l'utilisation des autorisations d'un niveau élevé aux seules fois où elles sont requises pour des tâches spécifiques. Grâce aux rôles, vous pouvez contribuer à prévenir les modifications accidentelles des environnements sensibles, en particulier si vous les associez à un audit pour vous assurer que les rôles ne sont utilisés qu'en cas de besoin.

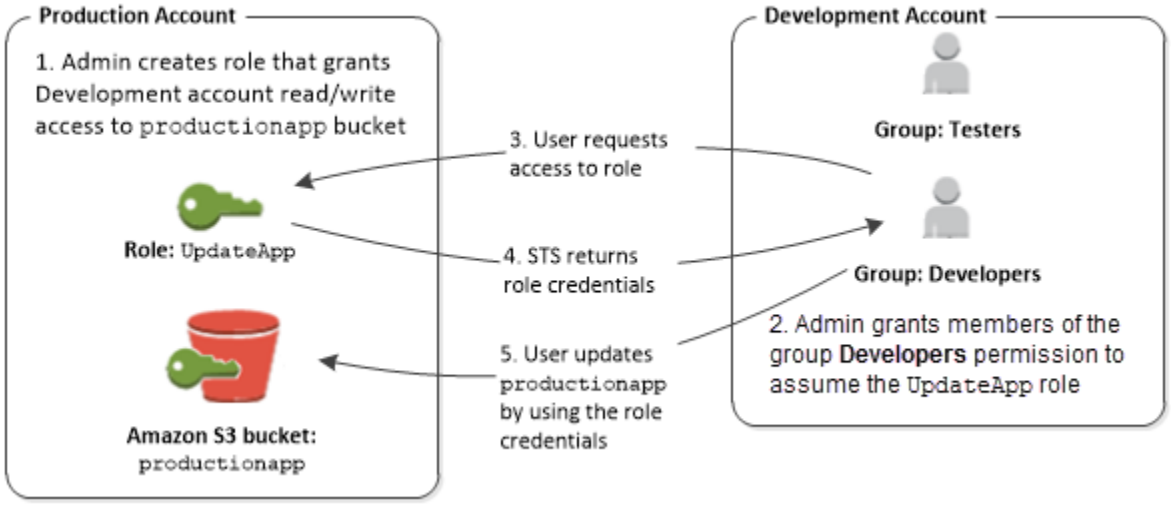
Lorsque vous créez un rôle à cette fin, vous spécifiez les comptes en fonction de l'ID des utilisateurs ayant besoin d'accéder à l'élément `Principal` de la politique d'approbation du rôle. Vous pouvez

ensuite accorder à des utilisateurs spécifiques de ces autres comptes des autorisations à passer au rôle. Pour savoir si les principaux des comptes situés en dehors de votre zone de confiance (organisation ou compte de confiance) ont accès à vos rôles, consultez Qu'est-ce qu'IAM Access Analyzer ?.

Un utilisateur d'un compte peut passer à un rôle dans le même compte ou dans un autre compte. Lorsqu'il utilise le rôle, l'utilisateur peut exécuter uniquement les actions et accéder uniquement aux ressources autorisées par le rôle. Leurs autorisations utilisateur d'origine sont suspendues. Lorsque l'utilisateur quitte le rôle, ses autorisations utilisateur d'origine sont restaurées.

Exemple de scénario utilisant des comptes de développement et de production distincts

Imaginez que votre organisation en dispose de plusieurs Comptes AWS pour isoler un environnement de développement d'un environnement de production. Les utilisateurs du compte de développement peuvent parfois avoir besoin d'accéder aux ressources du compte de production. Par exemple, vous pouvez avoir besoin d'un accès entre comptes lorsque vous promouvez une mise à jour depuis l'environnement de développement vers l'environnement de production. Même si vous pouvez créer des identités (et mots de passe) distinctes pour les utilisateurs qui travaillent dans les deux comptes, la gestion des informations d'identification pour plusieurs comptes rend la gestion des identités difficile. Dans l'illustration suivante, tous les utilisateurs sont gérés dans le compte de développement, mais certains développeurs nécessitent un accès limité au compte de production. Le compte de développement se compose de deux groupes : les Développeurs et les Testeurs, chacun disposant de sa propre politique.



1. Dans le compte de production, un administrateur utilise IAM pour créer le rôle `UpdateApp` dans ce compte. Dans le rôle, l'administrateur définit une politique d'approbation qui spécifie le compte

de développement en tant que `Principal`, ce qui signifie que les utilisateurs autorisés du compte de développement peuvent utiliser le rôle `UpdateApp`. L'administrateur définit également une politique d'autorisations pour le rôle qui spécifie les autorisations de lecture et d'écriture sur le compartiment Amazon S3 appelé `productionapp`.

L'administrateur partage ensuite les informations appropriées avec toute personne qui doit endosser le rôle. Ces informations sont le numéro de compte et le nom du rôle (pour les utilisateurs de la AWS console) ou le nom de ressource Amazon (ARN) (pour AWS CLI AWS l'accès à l'API). L'ARN du rôle peut ressembler à `arn:aws:iam::123456789012:role/UpdateApp`, où le rôle est appelé `UpdateApp`. Le rôle a été créé dans le compte dont le numéro est 123456789012.

> **Note**
>
> L'administrateur peut, facultativement, configurer le rôle afin que les utilisateurs qui endossent le rôle doivent d'abord être authentifiés à l'aide de l'authentification multi-facteur (MFA). Pour plus d'informations, veuillez consulter Accès sécurisé aux API avec MFA.

2. Dans le compte de développement, un administrateur accorde aux membres du groupe Développeurs l'autorisation de changer de rôle. Cela se fait en accordant au groupe de développeurs l'autorisation d'appeler l'`AssumeRole`API AWS Security Token Service (AWS STS) pour le `UpdateApp` rôle. Tout utilisateur IAM appartenant au groupe Développeurs dans le compte de développement peut à présent basculer vers le rôle `UpdateApp` du compte de production. Les autres utilisateurs n'appartenant pas au groupe de développeurs n'ont pas l'autorisation de passer au rôle et ne peuvent, donc, pas accéder au compartiment S3 dans le compte de production.
3. L'utilisateur demande à passer au rôle :
   - AWS console : l'utilisateur choisit le nom du compte dans la barre de navigation et choisit Switch Role. L'utilisateur spécifie l'ID (ou l'alias) du compte, ainsi que le nom du rôle. L'utilisateur peut également choisir un lien envoyé par e-mail par l'administrateur. Le lien dirige l'utilisateur vers la page Changer de rôle avec les détails déjà remplis.
   - AWS API/AWS CLI : Un utilisateur du groupe Développeurs du compte de développement appelle la `AssumeRole` fonction pour obtenir des informations d'identification pour le `UpdateApp` rôle. L'utilisateur spécifie l'ARN du rôle `UpdateApp` dans le cadre de l'appel. Si un utilisateur du groupe Testeurs fait la même demande, la demande échoue, car les Testeurs ne sont pas autorisés à appeler `AssumeRole` pour l'ARN de rôle `UpdateApp`.
4. AWS STS renvoie des informations d'identification temporaires :

- AWS console : AWS STS vérifie la demande avec la politique de confiance du rôle pour s'assurer que la demande provient d'une entité de confiance (il s'agit du compte de développement). Après vérification, AWS STS renvoie les informations d'identification de sécurité temporaires à la AWS console.
- API/CLI: AWS STS vérifie la demande par rapport à la politique de confiance du rôle pour s'assurer qu'elle provient d'une entité de confiance (il s'agit du compte de développement). Après vérification, AWS STS renvoie les informations de sécurité temporaires à l'application.

5. Les informations d'identification temporaires permettent d'accéder à la AWS ressource :
   - AWS console : la AWS console utilise les informations d'identification temporaires au nom de l'utilisateur pour toutes les actions ultérieures de la console, dans ce cas, pour lire et écrire dans le `productionapp` compartiment. La console ne peut pas accéder à d'autres ressources du compte de production. Lorsque l'utilisateur quitte le rôle, les autorisations dont ils disposait avant de changer de rôle sont restaurées.
   - API/CLI: L'application utilise les informations d'identification de sécurité temporaires pour mettre à jour le `productionapp` compartiment. Avec les informations d'identification de sécurité temporaires, l'application peut uniquement lire et écrire dans le compartiment `productionapp`, mais ne peut pas accéder à d'autres ressources du compte Production. L'application n'a pas besoin de quitter le rôle. Au contraire, elle arrête d'utiliser les informations d'identification temporaires et utilise les informations d'identification d'origine dans les appels d'API suivants.

Ressources supplémentaires

Pour plus d'informations, consultez les ressources suivantes :

- Tutoriel IAM : accès délégué via AWS comptes utilisant des rôles IAM

Accès pour les applications autres que les charges AWS de travail

Un rôle IAM est un objet dans Gestion des identités et des accès AWS (IAM) auquel des autorisations sont attribuées. Lorsque vous assumez ce rôle en utilisant une identité IAM ou une identité extérieure AWS, cela vous fournit des informations d'identification de sécurité temporaires pour votre session de rôle. Il se peut que des charges de travail s'exécutent dans votre centre de données ou dans d'autres infrastructures extérieures AWS qui doivent accéder à vos AWS ressources. Au lieu de créer, de distribuer et de gérer des clés d'accès à long terme, vous pouvez utiliser Gestion des identités et des accès AWS Roles Anywhere (IAM Roles Anywhere) pour authentifier vos autres charges de travail.AWS IAM Roles Anywhere utilise les certificats X.509 de

votre autorité de certification (CA) pour authentifier les identités et fournir un accès sécurisé à l' Services AWS aide des informations d'identification temporaires fournies par un rôle IAM.

Pour utiliser Rôles Anywhere IAM

1. Configurez une autorité de certification à l’aide de Autorité de certification privée AWS ou utilisez une autorité de certification issue de votre propre infrastructure PKI.
2. Après avoir configuré une autorité de certification, vous créez un objet dans Rôles Anywhere IAM appelé ancre d’approbation. Ce point d’ancrage établit une approbation entre Rôles Anywhere IAM et votre autorité de certification pour l’authentification.
3. Vous pouvez ensuite configurer vos rôles IAM existants ou créer de nouveaux rôles qui font confiance au service IAM Roles Anywhere.
4. Authentifiez vos tâches non liées aux AWS charges de travail avec IAM Roles Anywhere à l'aide de l'ancre de confiance. AWS accorde les informations d'identification temporaires non liées à la AWS charge de travail au rôle IAM qui a accès à vos AWS ressources.

Ressources supplémentaires

Les ressources suivantes peuvent vous aider à en savoir plus sur la fourniture d’un accès aux charges de travail non AWS .

- Pour plus d'informations sur IAM Roles Anywhere, consultez Présentation de Gestion des identités et des accès AWS Rôle Anywhere dans le Guide de l'utilisateur d'IAM Roles Anywhere.
- Pour savoir comment configurer une infrastructure à clé publique (PKI) pour Rôles Anywhere IAM, consultez la section Rôles Anywhere IAM avec une autorité de certification externe sur le Blog de sécuritéAWS .

Accès à Comptes AWS appartenant à des tiers

Lorsque des tiers ont besoin d'accéder aux AWS ressources de votre organisation, vous pouvez utiliser des rôles pour leur déléguer l'accès. Par exemple, un tiers peut fournir un service de gestion de vos ressources AWS . Grâce aux rôles IAM, vous pouvez autoriser ces tiers à accéder à vos AWS ressources sans partager vos informations AWS de sécurité. Au lieu de cela, le tiers peut accéder à vos AWS ressources en assumant un rôle que vous créez dans votre Compte AWS. Pour savoir si les principaux des comptes situés en dehors de votre zone de confiance (organisation ou compte de confiance) ont accès à vos rôles, consultez Qu'est-ce qu'IAM Access Analyzer ?.

Les tiers doivent vous fournir les informations suivantes pour vous permettre de créer un rôle qu'ils peuvent endosser :

- L' Compte AWS identifiant du tiers. Vous spécifiez leur ID d' Compte AWS en tant que principal lorsque vous définissez la politique d'approbation pour le rôle.
- Un ID externe à attacher uniquement à ce rôle. L'ID externe peut être n'importe quel identifiant qui n'est connu que de vous et de la tierce partie. Par exemple, vous pouvez utiliser un ID de facture entre les tiers et vous-même, mais n'utilisez aucun élément pouvant être deviné, tels que le nom ou le numéro de téléphone du tiers. Vous devez spécifier cet ID lorsque vous définissez la politique d'approbation pour le rôle. Les tiers doivent fournir cette ID lorsqu'ils endosser le rôle.
- Les autorisations dont le tiers a besoin pour utiliser vos ressources AWS . Vous devez spécifier ces autorisations lors de la définition de la politique d'autorisation du rôle. Cette politique définit les actions que les utilisateurs tiers peuvent entreprendre et les ressources auxquelles ils peuvent accéder.

Après avoir créé le rôle, vous devez fournir l'Amazon Resource Name (ARN) du rôle au tiers. Il a besoin de l'ARN de votre rôle pour endosser le rôle.

> **Important**
>
> Lorsque vous autorisez des tiers à accéder à vos AWS ressources, ils peuvent accéder à toutes les ressources que vous spécifiez dans la politique. L'utilisation de vos ressources par ces tiers vous est facturée. Veillez à limiter leur utilisation de vos ressources de façon appropriée.

ID externe pour l'accès d'un tiers

Un identifiant externe permet à l'utilisateur qui assume le rôle d'affirmer les circonstances dans lesquelles il travaille. Il permet également au titulaire du compte d'autoriser que le rôle soit endossé uniquement dans des circonstances spécifiques. La fonction principale de l'ID externe consiste à traiter et à prévenir Le problème de l'adjoint confus.

> **Important**
>
> AWS ne traite pas l'identifiant externe comme un secret. Une fois que vous avez créé un secret, tel qu'une paire de clés d'accès ou un mot de passe AWS, vous ne pouvez plus le

consulter. L'ID externe d'un rôle peut être consulté par toute personne autorisée à consulter le rôle.

Quand est-il conseillé d'utiliser un ID externe ?

Utilisez un ID externe dans les situations suivantes :

- Vous êtes Compte AWS propriétaire et vous avez configuré un rôle pour un tiers qui accède Comptes AWS à d'autres rôles que le vôtre. Vous devez demander à ce tiers un ID externe qu'il inclura lorsqu'il endossera votre rôle. Ensuite, vous vérifiez cet ID externe dans la politique d'approbation de votre rôle. Ainsi, la tierce partie peut endosser votre rôle uniquement lorsqu'elle agit en votre nom.
- Vous êtes en position d'endosser des rôles pour le compte de différents clients comme Exemple Corp dans notre précédent scénario. Vous devez attribuer un ID externe unique à chaque client et leur demander de l'ajouter à leur politique d'approbation du rôle. Vous devez ensuite vous assurer de toujours inclure l'ID externe correct dans vos demandes pour endosser des rôles.

  Vous disposez probablement déjà d'un identifiant unique pour chacun de vos clients, et cet ID unique est suffisant pour être utilisé comme ID externe. L'ID externe n'est pas une valeur spéciale que vous devez créer de manière explicite, ou suivre séparément, juste à cette fin.

  Vous devez toujours spécifier l'ID externe dans vos appels d'API `AssumeRole`. De plus, lorsqu'un client vous fournit un ARN de rôle, vérifiez que vous pouvez endosser le rôle avec et sans l'ID externe correct. Si vous pouvez endosser le rôle sans l'ID externe approprié, ne stockez pas l'ARN du rôle du client dans votre système. Attendez que le client mette à jour la politique d'approbation du rôle pour demander l'ID externe. Ainsi, vous aidez vos clients à faire ce qu'il faut, ce qui vous permet de vous protéger tous les deux contre le problème du député confus.

Exemple de scénario utilisant un ID externe

Par exemple, supposons que vous décidiez de faire appel à une société tierce appelée Example Corp pour surveiller vos activités Compte AWS et vous aider à optimiser vos coûts. Afin de suivre vos dépenses quotidiennes, Example Corp doit accéder à vos AWS ressources. Example Corp surveille également de nombreux autres comptes AWS pour d'autres clients.

Ne donnez pas à Exemple Corp l'accès à un utilisateur IAM et à ses informations d'identification à long terme dans votre compte AWS . Utilisez plutôt un rôle IAM et ses informations d'identification

de sécurité temporaires. Un rôle IAM fournit un mécanisme permettant à un tiers d'accéder à vos AWS ressources sans avoir à partager des informations d'identification à long terme (telles qu'une clé d'accès utilisateur IAM).

Vous pouvez utiliser un rôle IAM pour établir une relation de confiance entre votre Compte AWS et le compte Exemple Corp. Une fois cette relation établie, un membre du compte Example Corp peut appeler l' AWS Security Token Service AssumeRoleAPI pour obtenir des informations d'identification de sécurité temporaires. Les membres d'Example Corp peuvent ensuite utiliser les informations d'identification pour accéder aux AWS ressources de votre compte.

> **Note**
>
> Pour plus d'informations sur les opérations d' AWS API AssumeRole et les autres opérations que vous pouvez appeler pour obtenir des informations d'identification de sécurité temporaires, consultezComparez les AWS STS informations d'identification.

Voici comment s'articule ce scénario :

1. Vous embauchez Example Corp qui crée un identifiant client unique pour vous. Ils vous fournissent cet identifiant client unique et son Compte AWS numéro. Vous avez besoin de ces informations pour créer un rôle IAM à l'étape suivante.

   > **Note**
   >
   > Example Corp peut utiliser la valeur de chaîne de son choix pour le ExternalId, à condition qu'elle soit unique pour chaque client. Il peut s'agir d'un numéro de compte client ou encore d'une chaîne aléatoire de caractères, à condition que chaque client ait une valeur différente. Elle n'est pas censée être « secrète ». Example Corp doit fournir la ExternalId valeur à chaque client. Ce qui compte, c'est qu'elle soit générée par Example Corp et non par ses clients afin de garantir que chaque ID externe est unique.

2. Vous vous connectez AWS et créez un rôle IAM qui permet à Example Corp d'accéder à vos ressources. Comme tout autre rôle IAM, celui-ci dispose de deux politiques, une politique d'autorisation et une politique d'approbation. La politique d'approbation du rôle spécifie qui peut endosser le rôle. Dans notre exemple de scénario, la politique spécifie le Compte AWS nombre de Example Corp comme étant le`Principal`. Cela permet aux identités de ce compte d'endosser le rôle. En outre, vous ajoutez un élément `Condition` à la politique d'approbation.

Cette `Condition` teste la clé de contexte `ExternalId` afin de s'assurer qu'elle correspond à l'ID client unique issu d'Exemple Corp. Exemple :

```
   "Principal": {"AWS": "Example Corp's Compte AWS ID"},
   "Condition": {"StringEquals": {"sts:ExternalId": "Unique ID Assigned by Example
Corp"}}
```

3. La politique d'autorisation du rôle spécifie ce que le rôle permet à un utilisateur de faire. Par exemple, vous pouvez spécifier que le rôle permet à un utilisateur de gérer uniquement vos ressources Amazon EC2 et Amazon RDS, mais pas vos utilisateurs ou groupes IAM. Dans notre exemple de scénario, vous utilisez la politique d'autorisation pour accorder à Example Corp un accès en lecture seule à toutes les ressources de votre compte.
4. Après avoir créé le rôle, vous devez fournir l'Amazon Resource Name (ARN) du rôle à Example Corp.
5. Lorsqu'Example Corp a besoin d'accéder à vos AWS ressources, un membre de l'entreprise appelle l' AWS `sts:AssumeRole`API. L'appel inclut l'ARN du rôle à assumer et le ExternalId paramètre correspondant à son identifiant client.

Si la demande provient d'une personne utilisant Example Corp Compte AWS, et si l'ARN du rôle et l'ID externe sont corrects, la demande aboutit. Il fournit ensuite des informations d'identification de sécurité temporaires qu'Example Corp peut utiliser pour accéder aux AWS ressources autorisées par votre rôle.

Autrement dit, quand une politique de rôle inclut un ID externe, tous ceux qui souhaitent endosser le rôle doivent non seulement être spécifiés comme principaux dans le rôle, mais également inclure l'ID externe correct.

Points clés des ID externes

- Dans un environnement multi-tenant où vous gérez plusieurs clients avec des AWS comptes différents, nous vous recommandons d'utiliser un identifiant externe par. Compte AWS Cet ID doit être une chaîne aléatoire générée par le tiers.
- Pour exiger que le tiers fournisse un ID externe lors de la prise en charge d'un rôle, mettez à jour la politique d'approbation du rôle avec l'ID externe de votre choix.
- Pour fournir un ID externe lorsque vous assumez un rôle, utilisez l' AWS API AWS CLI or pour assumer ce rôle. Pour plus d'informations, consultez l'opération de l'AssumeRoleAPI STS ou l'opération de l'interface de ligne de commande STS assume-role.

- La valeur `ExternalId` peut avoir un minimum de 2 caractères et un maximum de 1 224 caractères. La valeur doit être alphanumérique sans espaces. Elle peut également inclure les symboles suivants : signe plus (+), signe égal (=), virgule (,), point (.), arobase (@), deux points (:), barre oblique (/) et tiret (-).

Ressources supplémentaires

Les ressources suivantes peuvent vous aider à en savoir plus sur la fourniture d'un accès aux Comptes AWS appartenant à des tiers.

- Pour savoir comment autoriser d'autres personnes à effectuer des actions dans votre Compte AWS environnement, consultezCréation d'un rôle à l'aide de politiques d'approbation personnalisées.
- Pour savoir comment accorder l'autorisation de basculer vers un rôle, consultez Octroi d'autorisations à un utilisateur pour endosser un rôle
- Pour savoir comment créer et fournir des informations d'identification de sécurité temporaires à des utilisateurs de confiance, consultezAutorisations affectées aux informations d'identification de sécurité temporaires.

Accès à un AWS service

De nombreux AWS services nécessitent que vous utilisiez des rôles pour contrôler les accès auxquels ils peuvent accéder. Un rôle qu'un service endosse pour effectuer des actions en votre nom s'appelle un rôle de service. Lorsqu'un rôle de service remplit un objectif spécial pour un service, il peut être défini comme rôle lié à un service. Consultez la documentation AWS spécifique à chaque service pour vérifier s'il utilise des rôles et apprendre à attribuer un rôle au service afin qu'il l'utilise.

Pour plus de détails sur la création d'un rôle permettant de déléguer l'accès à un service proposé par AWS, consultezCréez un rôle pour déléguer des autorisations à un AWS web.

Accès aux utilisateurs authentifiés en externe (fédération d'identité)

Vos utilisateurs ont peut-être déjà des identités extérieures AWS, par exemple dans votre annuaire d'entreprise. Si ces utilisateurs doivent travailler avec AWS des ressources (ou utiliser des applications qui accèdent à ces ressources), ils ont également besoin d'informations d'identification AWS de sécurité. A l'aide d'un rôle IAM, vous pouvez définir des autorisations pour des utilisateurs dont l'identité est fédérée par votre organisation ou un fournisseur d'identité (IdP) tiers.

> **Note**
>
> En tant que bonne pratique de sécurité, nous vous recommandons de gérer l'accès des utilisateurs dans IAM Identity Center à l'aide de la fédération d'identité plutôt que de créer des utilisateurs IAM. Pour en savoir plus sur les situations spécifiques dans lesquelles un utilisateur IAM est nécessaire, veuillez consulter Quand créer un utilisateur IAM (au lieu d'un rôle).

Les ensembles d'autorisations IAM Identity Center fournissent des modèles de rôles IAM gérés par IAM Identity Center. Le gestionnaire d'accès aux comptes, une fonctionnalité IAM, vous permet d'attribuer vos propres rôles IAM personnalisés aux utilisateurs et aux groupes IAM Identity Center, en activant des fonctionnalités telles que des politiques de confiance personnalisées, des balises de rôle et des chemins de rôles configurables. Vous pouvez l'utiliser conjointement avec des ensembles d'autorisations ou seul.

Fédération d'utilisateurs d'une application mobile ou web à l'aide d'Amazon Cognito

Si vous créez une application mobile ou Web qui accède à AWS des ressources, l'application a besoin d'informations d'identification de sécurité pour pouvoir adresser des requêtes programmatiques à. AWS Pour la plupart des scénarios impliquant des applications mobiles, nous recommandons d'utiliser Amazon Cognito. Vous pouvez utiliser ce service avec le SDK AWS mobile pour iOS et le SDK AWS mobile pour Android et Fire OS afin de créer des identités uniques pour les utilisateurs et de les authentifier pour un accès sécurisé à vos ressources. AWS Amazon Cognito prend en charge les mêmes fournisseurs d'identité que ceux répertoriés dans la section suivante, ainsi que les identités authentifiées par le développeur et les accès non authentifiés (invité). Amazon Cognito fournit également des opérations d'API pour la synchronisation des données utilisateur afin de les conserver à mesure que les utilisateurs passent d'un périphérique à l'autre. Pour plus d'informations, veuillez consulter Amazon Cognito pour les applications mobiles.

Fédération d'utilisateurs à l'aide de fournisseurs d'identité publics ou d'OpenID Connect

Dans la mesure du possible, utilisez Amazon Cognito pour les scénarios d'applications mobiles et Web. Amazon Cognito effectue pour vous la plupart des tâches en coulisses avec les services de fournisseur d'identité publique. Il fonctionne avec les mêmes services tiers et prend également en charge les connexions anonymes. Toutefois, dans le cas de scénarios plus complexes, vous pouvez avoir directement recours à un service tiers tel que Login with Amazon, Facebook, Google, ou tout

autre IdP compatible avec OpenID Connect (OIDC). Pour plus d'informations sur l'utilisation de la fédération OIDC avec l'un de ces services, consultez Fédération OIDC.

Fédération d'utilisateurs avec SAML 2.0

Si votre organisation utilise déjà un progiciel de fournisseur d'identité qui prend en charge le SAML 2.0 (Security Assertion Markup Language 2.0), vous pouvez créer une relation de confiance entre votre organisation en tant que fournisseur d'identité (IdP) et en AWS tant que fournisseur de services. Vous pouvez ensuite utiliser SAML pour fournir à vos utilisateurs une connexion unique fédérée (SSO) Console de gestion AWS ou un accès fédéré pour appeler les opérations de l'API. AWS Par exemple, si votre entreprise utilise Microsoft Active Directory et Active Directory Federation Services, vous pouvez utiliser la fédération à l'aide de SAML 2.0. Pour plus d'informations sur l'utilisation des utilisateurs fédérés avec SAML 2.0, consultez Fédération SAML 2.0.

Fédération d'utilisateurs via la création d'une application de broker d'identité personnalisé

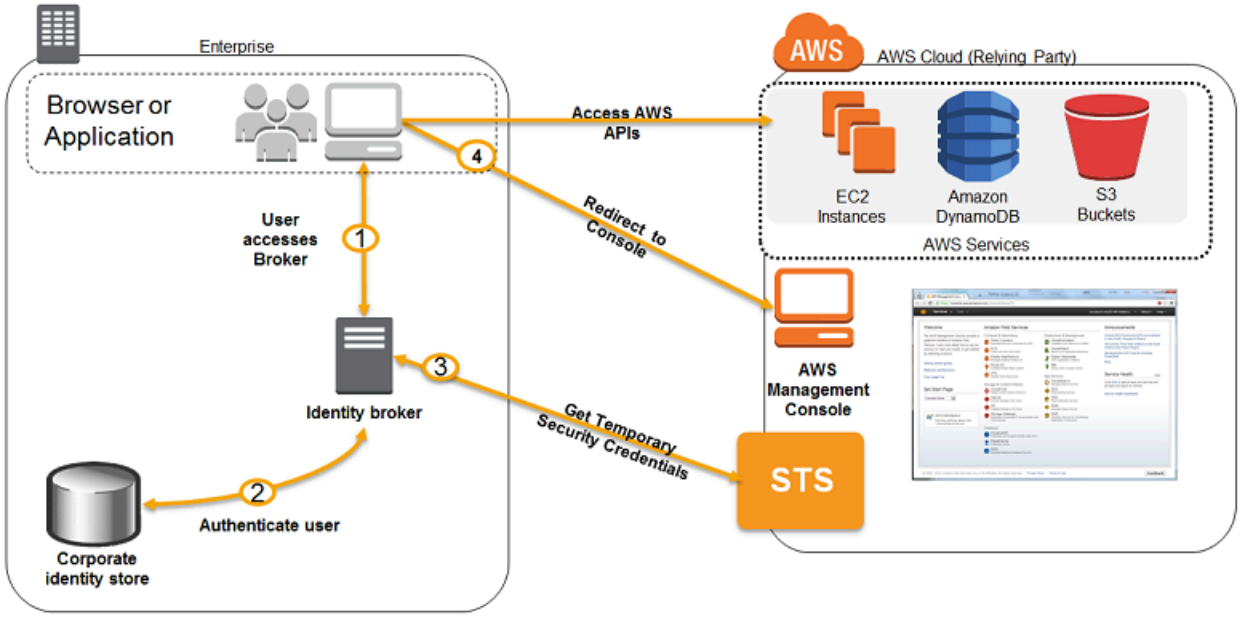
Si votre base d'identités n'est pas compatible avec SAML 2.0, vous pouvez créer une application de broker d'identité personnalisé capable d'exécuter une fonction similaire. L'application broker authentifie les utilisateurs, demande des informations d'identification temporaires pour les utilisateurs AWS, puis les fournit à l'utilisateur pour qu'il accède aux AWS ressources.

Par exemple, Example Corp. compte de nombreux employés qui doivent exécuter des applications internes qui accèdent aux AWS ressources de l'entreprise. Ils disposent déjà d'identités dans le système d'identité et d'authentification de l'entreprise et, par conséquent, Example Corp. ne souhaite pas créer un utilisateur IAM séparé pour chacun de ses employés.

Bob est développeur chez Example Corp. Pour permettre aux applications internes d'Example Corp. d'accéder aux AWS ressources de l'entreprise, Bob développe une application de courtier d'identité personnalisée. L'application vérifie que les employés sont connectés au système d'identité et d'authentification existant d'Example Corp., par exemple LDAP, Active Directory ou un autre système. L'application de broker d'identité obtient ensuite des informations d'identification de sécurité temporaires pour les employés. Ce scénario est similaire au précédent (une application mobile utilisant un système d'authentification personnalisé), sauf que les applications qui ont besoin d'accéder aux AWS ressources s'exécutent toutes au sein du réseau de l'entreprise et que l'entreprise dispose d'un système d'authentification existant.

Pour obtenir les informations d'identification de sécurité temporaires, l'application de broker d'identité appelle `AssumeRole` ou `GetFederationToken`, selon la façon dont Bob veut gérer les politiques des utilisateurs et le délai d'expiration des informations d'identification. Pour plus d'informations sur les différences entre ces opérations d'API, consultez Informations d'identification de sécurité

temporaires dans IAM et Autorisations affectées aux informations d'identification de sécurité temporaires.) L'appel renvoie des informations d'identification de sécurité temporaires composées d'un identifiant de clé d' AWS accès, d'une clé d'accès secrète et d'un jeton de session. L'application de broker d'identité rend ensuite ces informations d'identification de sécurité temporaires accessibles à l'application interne de l'entreprise. L'application peut alors utiliser ces informations d'identification de sécurité temporaires pour appeler directement AWS . L'application met en cache les informations d'identification jusqu'à ce qu'elles parviennent à expiration, puis demande un nouvel ensemble d'informations d'identification temporaires. L'illustration suivante décrit ce scénario.



Ce scénario utilise les attributs suivants :

- L'application de broker d'identité dispose d'autorisations d'accès à l'API du service de jeton (STS) d'IAM pour créer des informations d'identification de sécurité temporaires.
- L'application de broker d'identité peut vérifier que les employés sont authentifiés dans le système d'authentification existant.
- Les utilisateurs peuvent obtenir une URL temporaire qui leur donne accès à la console de AWS gestion (appelée authentification unique).

Pour plus d'informations sur la création d'informations d'identification de sécurité temporaires, consultez Comparez les AWS STS informations d'identification. Pour plus d'informations sur l'accès des principaux fédérés SAML à la console de AWS gestion, consultez. Permettre aux principaux fédérés SAML 2.0 d'accéder au Console de gestion AWS

Création d’un rôle IAM

Pour créer un rôle, vous pouvez utiliser les Console de gestion AWS AWS CLI outils pour Windows ou PowerShell l'API IAM.

Si vous utilisez le Console de gestion AWS, un assistant vous guide tout au long des étapes de création d'un rôle. L'assistant comporte des étapes légèrement différentes selon que vous créez un rôle pour un AWS service, pour un Compte AWS principal fédéré SAML ou OIDC.

Rôles pour les utilisateurs IAM

Créez ce rôle pour déléguer des autorisations au sein de votre rôle Compte AWS ou à des rôles définis dans d'autres rôles Comptes AWS que vous possédez. Un utilisateur d'un compte peut passer à un rôle dans le même compte ou dans un autre compte. Lorsqu'il utilise le rôle, l'utilisateur peut exécuter uniquement les actions et accéder uniquement aux ressources autorisées par le rôle. Leurs autorisations utilisateur d'origine sont suspendues. Lorsque l'utilisateur quitte le rôle, ses autorisations utilisateur d'origine sont restaurées.

Pour de plus amples informations, veuillez consulter Créer un rôle pour attribuer des autorisations à un utilisateur IAM.

Pour de plus amples informations sur la création de rôles avec un accès intercompte, consultez Création d’un rôle à l’aide de politiques d’approbation personnalisées.

Rôles pour AWS services

Créez ce rôle pour déléguer des autorisations à un service qui peut effectuer des actions en votre nom. Un rôle de service que vous transmettez à un service doit avoir une politique IAM avec les autorisations qui permettent au service d’effectuer les actions associées à ce service. Différentes autorisations sont requises pour chaque service AWS .

Pour plus d’informations sur la création de rôles de service, consultez Créez un rôle pour déléguer des autorisations à un AWS web.

Pour plus d’informations sur la création de rôles liés à un service, consultez Créer un rôle lié à un service.

Rôles pour la fédération d’identité

Créez ce rôle pour déléguer des autorisations aux utilisateurs qui ont déjà des identités en dehors d’ AWS. Lorsque vous utilisez un fournisseur d'identité , vous n'avez pas à créer de code de connexion personnalisé ni à gérer vos propres identités utilisateur. Vos utilisateurs externes se connectent via

un IdP et vous pouvez autoriser ces identités externes à utiliser les AWS ressources de votre compte. Les fournisseurs d'identité contribuent à la sécurité de votre AWS compte, car vous n'avez pas à distribuer ou à intégrer des informations de sécurité à long terme, telles que des clés d'accès, dans votre application.

Pour de plus amples informations, veuillez consulter Créer un rôle pour un fournisseur d'identité tiers.

Créer un rôle pour attribuer des autorisations à un utilisateur IAM

Vous pouvez utiliser les rôles IAM pour accéder à vos AWS ressources. Avec les rôles IAM, vous pouvez établir des relations de confiance entre votre compte de confiance et d'autres comptes de AWS confiance. Le compte d'approbation est propriétaire des ressources auxquelles les utilisateurs ont accès , tandis que le compte approuvé contient les utilisateurs devant accéder aux ressources. Toutefois, il est possible qu'un autre compte détienne une ressource dans votre compte. Par exemple, le compte de confiance peut autoriser le compte approuvé à créer de nouvelles ressources, telles que la création de nouveaux objets dans un compartiment Amazon S3. Dans ce cas, le compte qui crée la ressource la détient et contrôle les personnes pouvant y accéder.

Une fois que vous avez créé la relation de confiance, un utilisateur IAM ou une application du compte sécurisé peut utiliser l'opération d'`AssumeRole`API AWS Security Token Service (AWS STS). Cette opération fournit des informations de sécurité temporaires qui permettent d'accéder aux AWS ressources de votre compte.

Les deux comptes peuvent être contrôlés par vous-même, ou le compte contenant les utilisateurs peut être contrôlé par un tiers. Si l'autre compte associé aux utilisateurs est un compte Compte AWS que vous ne contrôlez pas, vous pouvez utiliser l'`externalId`attribut. L'ID externe peut être n'importe quel mot ou nombre convenu avec l'administrateur du compte tiers. Cette option ajoute automatiquement une condition à la politique d'approbation. Cette condition permet à l'utilisateur d'endosser le rôle uniquement si la demande inclut l'élément approprié `sts:ExternalID`. Pour de plus amples informations, veuillez consulter Accès à Comptes AWS appartenant à des tiers.

Pour plus d'informations sur la façon d'utiliser les rôles pour déléguer des autorisations, consultez Termes et concepts relatifs aux rôles. Pour en savoir plus sur l'utilisation d'un rôle de service afin de permettre aux services l'accès aux ressources de votre compte, consultez Créez un rôle pour déléguer des autorisations à un AWS web.

Création d'un rôle IAM (console)

Vous pouvez utiliser le Console de gestion AWS pour créer un rôle qu'un utilisateur IAM peut assumer. Supposons, par exemple, que votre organisation en dispose Comptes AWS de

plusieurs pour isoler un environnement de développement d'un environnement de production. Pour des informations générales sur la création d'un rôle autorisant les utilisateurs du compte de développement à accéder aux ressources du compte de production, consultez Exemple de scénario utilisant des comptes de développement et de production distincts.

Autorisations minimales

Pour effectuer les étapes suivantes, vous devez au moins disposer des autorisations IAM suivantes :

- `access-analyzer:ValidatePolicy`
- `iam:AttachRolePolicy`
- `iam:CreatePolicy`
- `iam:CreateRole`
- `iam:GetAccountSummary`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:GetRole`
- `iam:ListAccountAliases`
- `iam:ListAttachedRolePolicies`
- `iam:ListOpenIDConnectProviders`
- `iam:ListPolicies`
- `iam:ListRolePolicies`
- `iam:ListRoles`
- `iam:ListRoleTags`
- `iam:ListSAMLProviders`

Console

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de la console, choisissez Rôles, puis Créer un rôle.
3. Choisissez le type du rôle de l'Compte AWS.

4. Pour créer un rôle pour votre compte, choisissez This account (Ce compte). Pour créer un rôle pour un autre compte, choisissez Another Compte AWS (Autre) et saisissez l'Account ID (ID de compte) auquel vous voulez octroyer l'accès à vos ressources.

   L'administrateur du compte spécifié peut accorder l'autorisation d'endosser ce rôle à n'importe quel utilisateur IAM de ce compte. Pour ce faire, l'administrateur attache une politique à l'utilisateur ou à un groupe qui donne l'autorisation pour l'action `sts:AssumeRole`. Cette politique doit spécifier l'ARN du rôle comme `Resource`.

5. Si vous accordez des autorisations aux utilisateurs d'un compte que vous ne contrôlez pas et si les utilisateurs ont l'intention d'assumer ce rôle par programmation, alors sélectionnez Require external ID (Demander un ID externe). L'ID externe peut être n'importe quel mot ou nombre convenu avec l'administrateur du compte tiers. Cette option ajoute automatiquement une condition à la politique d'approbation. Cette condition permet à l'utilisateur d'endosser le rôle uniquement si la demande inclut l'élément approprié `sts:ExternalID`. Pour de plus amples informations, veuillez consulter Accès à Comptes AWS appartenant à des tiers.

   > **Important**
   >
   > Le choix de cette option restreint l'accès au rôle uniquement par le biais AWS CLI des outils pour Windows PowerShell ou de l' AWS API. Cela est dû au fait que vous ne pouvez pas utiliser la AWS console pour passer à un rôle dont la politique de confiance est assortie d'une `externalId` condition. Vous pouvez néanmoins créer ce type d'accès par programmation, en écrivant un script ou une application à l'aide du kit SDK approprié. Pour plus d'informations et un exemple de script, consultez How to Enable Cross-Account Access to the Console de gestion AWS sur AWS Security Blog.

6. Si vous souhaitez restreindre le rôle aux utilisateurs qui se connectent via une authentification MFA, sélectionnez Demander l'authentification MFA. Cela ajoute une condition à la politique d'approbation du rôle qui exige une authentification MFA. Un utilisateur qui veut endosser le rôle doit se connecter avec un mot de passe unique temporaire à partir d'un dispositif MFA configuré. Sans authentification MFA, les utilisateurs ne peuvent pas endosser le rôle. Pour plus d'informations sur l'authentification MFA, consultez AWS Multi-factor authentification dans IAM
7. Choisissez Suivant.
8. IAM inclut une liste des politiques AWS gérées et gérées par le client dans votre compte. Sélectionnez la politique à utiliser pour la politique d'autorisations ou choisissez Créer une

politique pour ouvrir un nouvel onglet de navigateur et créer une nouvelle politique de bout en bout. Pour de plus amples informations, veuillez consulter Création de politiques IAM. Une fois la politique créée, fermez cet onglet et revenez à l'onglet initial. Cochez la case en regard des politiques d'autorisations que vous souhaitez octroyer à toute personne endossant le rôle. Si vous préférez, vous pouvez ne sélectionner aucune stratégie pour le moment, puis les attacher au rôle ultérieurement. Par défaut, un rôle ne dispose d'aucune autorisation.

9. (Facultatif) Définissez une limite d'autorisations. Il s'agit d'une fonctionnalité avancée.

   Ouvrez la section Set permissions boundary (Définir une limite d'autorisations) et choisissez Use a permissions boundary to control the maximum role permissions (Utiliser une limite d'autorisations pour contrôler le nombre maximum d'autorisations de rôle). Sélectionnez la politique à utiliser comme limite d'autorisations.

10. Choisissez Suivant.
11. Dans le champ Role name (Nom de rôle), saisissez un nom pour votre rôle. Les noms de rôles doivent être uniques au sein de votre Compte AWS. Lorsqu'un nom de rôle est utilisé dans une politique ou dans le cadre d'un ARN, il est sensible à la casse. Lorsque le nom d'un rôle apparaît aux clients dans la console, par exemple lors du processus de connexion, il n'est pas sensible à la casse. Différentes entités peuvent référencer le rôle et il n'est donc pas possible de modifier son nom après sa création.
12. (Facultatif) Pour Description, saisissez une description pour le nouveau rôle.
13. Choisissez Edit (Modifier) dans les sections Step 1: Select trusted entities (Étape 1 : sélection d'entités de confiance) ou Step 2: Add permissions (Étape 2 : ajouter des autorisations) pour modifier les cas d'utilisation et les autorisations pour le rôle. Vous serez renvoyé aux pages précédentes pour effectuer les modifications.
14. (Facultatif) Ajoutez des métadonnées au rôle en associant les identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Tags pour Gestion des identités et des accès AWS resources.
15. Passez en revue les informations du rôle, puis choisissez Créer un rôle.

> **Important**
>
> N'oubliez pas que ceci ne représente que la première moitié de la configuration requise. Vous devez également accorder aux utilisateurs individuels du compte approuvé des autorisations permettant de basculer vers le rôle dans la console ou

d'endosser le rôle par programmation. Pour plus d'informations sur cette étape, consultez Octroi d'autorisations à un utilisateur pour endosser un rôle.

Création d'un rôle IAM (AWS CLI)

La création d'un rôle à partir de AWS CLI implique plusieurs étapes. Lorsque vous utilisez la console pour créer un rôle, la plupart des étapes sont effectuées pour vous, mais AWS CLI vous devez effectuer chaque étape vous-même de manière explicite. Vous devez créer le rôle et lui attribuer une politique d'autorisations. Vous pouvez également définir la limite d'autorisations pour votre rôle.

Pour créer un rôle pour l'accès entre comptes (AWS CLI)

1. Créez un rôle : aws iam create-role
2. Associez une politique d'autorisations gérées au rôle : aws iam attach-role-policy

   or

   Créez une politique d'autorisation intégrée pour le rôle : aws iam put-role-policy
3. (Facultatif) Ajoutez des attributs personnalisés au rôle en associant des balises : aws iam tag-role

   Pour de plus amples informations, veuillez consulter Gestion des balises sur les rôles IAM (AWS CLI or AWS API).
4. (Facultatif) Définissez la limite des autorisations pour le rôle : aws iam put-role-permissions-boundary

   Une limite d'autorisations contrôle les autorisations maximum dont un rôle peut disposer. Les limites d'autorisations constituent une AWS fonctionnalité avancée.

L'exemple suivant illustre les deux premières étapes les plus courantes pour créer un rôle entre comptes dans un environnement simple. Cet exemple permet à tout utilisateur du compte `123456789012` d'endosser le rôle et d'afficher le compartiment Amazon S3 `example_bucket`. Cet exemple suppose également que vous utilisiez un ordinateur client exécutant Windows et que vous ayez déjà configuré votre interface de ligne de commande à l'aide des informations d'identification et de la région de votre compte. Pour plus d'informations, voir Configuration de l'interface de ligne de AWS commande.

Dans cet exemple, incluez la politique de confiance suivante dans la première commande lors de la création du rôle. Cette politique de confiance permet aux utilisateurs du compte `123456789012` d'endosser le rôle à l'aide de l'opération `AssumeRole`, mais uniquement s'ils fournissent l'authentification MFA à l'aide des paramètres `SerialNumber` et `TokenCode`. Pour plus d'informations sur l'authentification MFA, consultez AWS Multi-factor authentification dans IAM.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
          "Action": "sts:AssumeRole",
          "Condition": { "Bool": { "aws:MultiFactorAuthPresent": "true" } }
      }
  ]
}
```

Important

Si votre élément `Principal` contient l'ARN d'un rôle ou utilisateur IAM spécifique, alors cet ARN devient un ID du principal unique lorsque la politique est enregistrée. Cela permet de réduire le risque d'escalade des autorisations par la suppression et la nouvelle création du rôle ou de l'utilisateur. Cet ID n'est pas fréquent dans la console, car il existe également une transformation inverse, pour revenir à l'ARN, lorsque la politique d'approbation est affichée. Toutefois, si vous supprimez le rôle ou l'utilisateur, l'ID principal apparaît dans la console car il n'est plus AWS possible de le mapper à un ARN. Par conséquent, si vous supprimez et recréez un utilisateur ou rôle référencé dans l'élément `Principal` d'une politique de confiance, vous devez modifier le rôle afin de remplacer l'ARN.

Lorsque vous utilisez la deuxième commande, vous devez attacher au rôle une politique gérée existante. La politique d'autorisations suivante permet à toute personne endossant le rôle d'exécuter uniquement l'action `ListBucket` sur le compartiment Amazon S3 `example_bucket`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Action": "s3:ListBucket",
          "Resource": "arn:aws:s3:::example_bucket"
      }
  ]
}
```

Pour créer ce rôle `Test-UserAccess-Role`, vous devez d'abord enregistrer la précédente politique de confiance avec le nom `trustpolicyforacct123456789012.json` dans le dossier `policies` de votre disque local `C:`. Enregistrez ensuite la politique d'autorisation précédente en tant que politique gérée par le client dans votre Compte AWS nom`PolicyForRole`. Vous pouvez ensuite utiliser les commandes suivantes pour créer le rôle et attacher la politique gérée.

```
# Create the role and attach the trust policy file that allows users in the specified
 account to assume the role.
$ aws iam create-role --role-name Test-UserAccess-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforacct123456789012.json

# Attach the permissions policy (in this example a managed policy) to the role to
 specify what it is allowed to do.
$ aws iam attach-role-policy --role-name Test-UserAccess-Role --policy-arn
 arn:aws:iam::123456789012:policy/PolicyForRole
```

> **Important**
>
> N'oubliez pas que ceci ne représente que la première moitié de la configuration requise. Vous devez également accorder à des utilisateurs individuels du compte approuvé les autorisations permettant de changer de rôle. Pour plus d'informations sur cette étape, consultez Octroi d'autorisations à un utilisateur pour endosser un rôle.

Une fois que vous avez créé le rôle et que vous lui avez accordé les autorisations nécessaires pour effectuer des AWS tâches ou accéder aux AWS ressources, tous les utilisateurs du `123456789012` compte peuvent assumer le rôle. Pour de plus amples informations, veuillez consulter Passez à un rôle IAM (AWS CLI).

Création d'un rôle IAM (AWS API)

La création d'un rôle à partir de l' AWS API implique plusieurs étapes. Lorsque vous utilisez la console pour créer un rôle, la plupart des étapes sont exécutées automatiquement pour vous, mais avec l'API vous devez exécuter explicitement chaque étape vous-même. Vous devez créer le rôle et lui attribuer une politique d'autorisations. Vous pouvez également définir la limite d'autorisations pour votre rôle.

Pour créer un rôle dans le code (AWS API)

1. Créez un rôle : CreateRole

   Vous pouvez spécifier un emplacement de fichier pour la politique d'approbation du rôle.
2. Associez une politique d'autorisation gérée au rôle : AttachRolePolicy

   or

   Créez une politique d'autorisation intégrée pour le rôle : PutRolePolicy

   > **Important**
   >
   > N'oubliez pas que ceci ne représente que la première moitié de la configuration requise. Vous devez également accorder à des utilisateurs individuels du compte approuvé les autorisations permettant de changer de rôle. Pour plus d'informations sur cette étape, consultez Octroi d'autorisations à un utilisateur pour endosser un rôle.
3. (Facultatif) Ajoutez des attributs personnalisés au rôle en attachant des balises : TagRole

   Pour de plus amples informations, veuillez consulter Gestion des balises sur les rôles IAM (AWS CLI or AWS API).
4. (Facultatif) Définissez la limite des autorisations pour le rôle : PutRolePermissionsBoundary

   Une limite d'autorisations contrôle les autorisations maximum dont un rôle peut disposer. Les limites d'autorisations constituent une AWS fonctionnalité avancée.

Une fois que vous avez créé le rôle et que vous lui avez accordé les autorisations nécessaires pour effectuer des AWS tâches ou accéder aux AWS ressources, vous devez accorder des autorisations aux utilisateurs du compte pour leur permettre d'assumer le rôle. Pour plus d'informations sur l'endossement d'un rôle, consultez Passez à un rôle IAM (AWS API).

Création d'un rôle IAM (AWS CloudFormation)

Pour plus d'informations sur la création d'un rôle IAM dans AWS CloudFormation, consultez la référence sur les ressources et les propriétés et les exemples dans le guide de l'AWS CloudFormation utilisateur.

Pour plus d'informations sur les modèles IAM dans AWS CloudFormation, consultez les extraits de Gestion des identités et des accès AWS modèles dans le Guide de l'AWS CloudFormation utilisateur.

Créez un rôle pour déléguer des autorisations à un AWS web

De nombreux AWS services nécessitent que vous utilisiez des rôles pour permettre au service d'accéder aux ressources d'autres services en votre nom. Un rôle qu'un service endosse pour effectuer des actions en votre nom s'appelle un rôle de service. Lorsqu'un rôle de service remplit un objectif spécial pour un service, il est défini comme rôle lié à un service. Pour savoir quels services prennent en charge par les rôles de service liés à un service ou déterminer si un service prend en charge une forme d'informations d'identification temporaires, consultez AWS services qui fonctionnent avec IAM. Pour découvrir comment un service individuel utilise des rôles, choisissez son nom dans le tableau pour afficher la documentation de ce service.

Lors de la définition de l'autorisation `PassRole`, vous devez vous assurer qu'un utilisateur ne transfère pas un rôle ayant plus d'autorisations que ce que vous souhaitez pour l'utilisateur. Par exemple, Alice n'est peut-être pas autorisée à effectuer des actions Amazon S3. Si Alice pouvait transmettre un rôle à un service autorisant les actions Amazon S3, le service pourrait effectuer des actions Amazon S3 pour le compte d'Alice lors de l'exécution de la tâche.

Pour plus d'informations sur la façon dont les rôles aident à déléguer des autorisations, consultez Termes et concepts relatifs aux rôles.

Autorisations de fonction de service

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur ou un rôle) de créer ou modifier un rôle lié à un service.

> **Note**
>
> L'ARN d'un rôle lié à un service comprend un principal de service indiqué dans les politiques suivantes comme *SERVICE-NAME*`.amazonaws.com`. N'essayez pas de deviner le principal de service, car il est sensible à la casse et le format peut varier entre les services AWS . Pour afficher le principal de service d'un service, consultez la documentation de son rôle lié à un service.

Pour permettre à une entité IAM de créer un rôle spécifique lié à un service

Ajoutez la politique suivante à l'entité IAM qui doit créer le rôle lié à un service. Cette politique vous permet de créer un rôle lié au service spécifié et avec un nom spécifique. Vous pouvez ensuite attacher des politiques en ligne ou gérées à ce rôle.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:CreateRole",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        }
    ]
}
```

Pour permettre à une entité IAM de créer n"importe quel rôle lié à un service

AWS recommande de n'autoriser que les utilisateurs administratifs à créer un rôle de service. Une personne disposant des autorisations nécessaires pour créer un rôle et attacher n'importe quelle politique peut augmenter ses propres autorisations. Au lieu de cela, créez une politique qui l'autorise à créer uniquement les rôles dont elle a besoin ou de demander à un administrateur de créer la fonction du service en son nom.

Pour joindre une politique permettant à un administrateur d'accéder à l'intégralité de votre Compte AWS contenu, utilisez la politique AdministratorAccess AWS gérée.

Pour permettre à une entité IAM de modifier un rôle lié à un service

Ajoutez la politique suivante à l'entité IAM qui doit modifier le rôle lié à un service.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "EditSpecificServiceRole",
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:DeleteRolePolicy",
                "iam:DetachRolePolicy",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:ListAttachedRolePolicies",
                "iam:ListRolePolicies",
                "iam:PutRolePolicy",
                "iam:UpdateRole",
                "iam:UpdateRoleDescription"
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        },
        {
            "Sid": "ViewRolesAndPolicies",
            "Effect": "Allow",
            "Action": [
                "iam:GetPolicy",
                "iam:ListRoles"
            ],
            "Resource": "*"
        }
    ]
}
```

Pour permettre à une entité IAM de supprimer un rôle spécifique lié à un service

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit supprimer le rôle lié à un service spécifié.

```
{
    "Effect": "Allow",
    "Action": "iam:DeleteRole",
    "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
}
```

Pour permettre à une entité IAM de supprimer un rôle de service

AWS recommande de n'autoriser que les utilisateurs administratifs à supprimer un rôle de service. Au lieu de cela, créez une politique qui leur permet de supprimer uniquement les rôles dont ils ont besoin ou de demander à un administrateur de supprimer le rôle de service en leur nom.

Pour joindre une politique permettant à un administrateur d'accéder à l'intégralité de votre Compte AWS contenu, utilisez la politique AdministratorAccess AWS gérée.

Création d'un rôle pour AWS service (console)

Vous pouvez utiliser le Console de gestion AWS pour créer un rôle pour un service. Puisque certains services prennent en charge plus d'un rôle de service, consultez la documentation AWS de votre service pour savoir quel cas d'utilisation choisir. Vous pouvez apprendre à attribuer les politiques d'approbation et d'autorisation nécessaires au rôle, afin que le service puisse endosser le rôle à votre place. Les étapes que vous pouvez utiliser pour contrôler les autorisations de votre rôle peuvent varier, selon la façon dont le service définit les cas d'utilisation et si vous créez un rôle lié à un service.

Console

Pour créer un rôle pour Service AWS (console IAM)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le volet de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
3. Pour Trusted entity (Entité de confiance), choisissez Service AWS.
4. Pour Service ou cas d'utilisation, choisissez un service, puis choisissez le cas d'utilisation. Les cas d'utilisation sont définis par le service pour inclure la politique d'approbation nécessaire au service.

5. Choisissez Suivant.
6. Pour Politiques d'autorisations, les options dépendent du cas d'utilisation que vous avez sélectionné :

   - Si le service définit les autorisations pour le rôle, il n'est pas possible de sélectionner les politiques d'autorisation.
   - Choisissez parmi un ensemble limité de politiques d'autorisation.
   - Choisissez parmi toutes les politiques d'autorisation.
   - Ne sélectionnez aucune politique d'autorisation, créez les politiques une fois le rôle créé, puis attachez-les au rôle.
7. (Facultatif) Définissez une limite d'autorisations. Il s'agit d'une fonctionnalité avancée disponible pour les fonctions de service, mais pas pour les rôles liés à un service.

   a. Ouvrez la section Définir une limite des autorisations et choisissez Utiliser une limite des autorisations pour contrôler le nombre maximum d'autorisations de rôle.

      IAM inclut une liste des politiques AWS gérées et gérées par le client dans votre compte.

   b. Sélectionnez la politique à utiliser comme limite d'autorisations.
8. Choisissez Suivant.
9. Pour Nom du rôle, les options dépendent du service :

   - Si le service définit le nom du rôle, vous ne pouvez pas modifier le nom du rôle.
   - Si le service définit un préfixe pour le nom du rôle, vous pouvez saisir un suffixe facultatif.
   - Si le service ne définit pas le nom du rôle, vous pouvez le nommer.

   > **Important**
   >
   > Lorsque vous nommez un rôle, notez ce qui suit :
   >
   > - Les noms de rôles doivent être uniques au sein de votre Compte AWS organisation et ne peuvent pas être rendus uniques au cas par cas.
   >
   >   Par exemple, ne créez pas deux rôles nommés **`PRODROLE`** et **`prodrole`**. Lorsqu'un nom de rôle est utilisé dans une politique ou dans le cadre d'un ARN, le nom de rôle est sensible à la casse. Cependant, lorsqu'un nom de rôle apparaît aux clients dans la console, par exemple lors de la procédure d'ouverture de session, le nom de rôle est insensible à la casse.

- Vous ne pouvez pas modifier le nom du rôle après sa création, car d'autres entités pourraient y faire référence.

10. (Facultatif) Pour Description, saisissez la description du rôle.
11. (Facultatif) Pour modifier les cas d'utilisation et les autorisations du rôle, dans les sections Étape 1 : sélectionner les entités de confiance ou Étape 2 : ajouter des autorisations, sélectionnez Modifier.
12. (Facultatif) Pour identifier, organiser ou rechercher le rôle, ajoutez des identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, consultez la section Balises pour les Gestion des identités et des accès AWS ressources dans le Guide de l'utilisateur IAM.
13. Passez en revue les informations du rôle, puis choisissez Create role (Créer un rôle).

Création d'un rôle pour un service (AWS CLI)

La création d'un rôle à partir de AWS CLI implique plusieurs étapes. Lorsque vous utilisez la console pour créer un rôle, de nombreuses étapes sont effectuées pour vous, mais AWS CLI vous devez explicitement effectuer chaque étape vous-même. Vous devez créer le rôle et lui attribuer une politique d'autorisations. Si le service concerné est Amazon EC2 vous devez également créer un profil d'instance et lui ajouter le rôle. Vous pouvez également définir la limite d'autorisations pour votre rôle.

Pour créer un rôle pour AWS service du AWS CLI

1. La `create-role` commande suivante crée un rôle nommé Test-Role et y associe une politique de confiance :

   ```
   aws iam create-role --role-name Test-Role --assume-role-policy-document
   file://Test-Role-Trust-Policy.json
   ```
2. Attachez une politique d'autorisations gérée au rôle : aws iam attach-role-policy.

   Par exemple, la commande `attach-role-policy` suivante attache la politique gérée par AWS , nommée `ReadOnlyAccess`, au rôle IAM nommé `ReadOnlyRole` :

   ```
   aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
   ReadOnlyAccess --role-name ReadOnlyRole
   ```

   or

Créez une politique d'autorisations en ligne pour le rôle : aws iam put-role-policy

Pour ajouter une politique d'autorisations en ligne, reportez-vous à l'exemple suivant :

```
aws iam put-role-policy --role-name Test-Role --policy-name
ExamplePolicy --policy-document file://AdminPolicy.json
```

3. (Facultatif) Ajoutez des attributs personnalisés au rôle en associant des balises : aws iam tag-role

   Pour plus d'informations, veuillez consulter Gestion des balises sur les rôles IAM (AWS CLI or AWS API).
4. (Facultatif) Définissez la limite d'autorisations pour le rôle : aws iam put-role-permissions-boundary

   Une limite d'autorisations contrôle les autorisations maximum dont un rôle peut disposer. Les limites d'autorisations constituent une AWS fonctionnalité avancée.

Si vous comptez utiliser le rôle avec Amazon EC2 ou un autre AWS service utilisant Amazon EC2, vous devez stocker le rôle dans un profil d'instance. Un profil d'instance est un conteneur pour un rôle que vous pouvez attacher à une instance Amazon EC2 lorsqu'elle est lancée. Un profil d'instance peut contenir un rôle uniquement et cette limite ne peut pas être augmentée. Si vous créez le rôle à l'aide du Console de gestion AWS, le profil d'instance est créé pour vous avec le même nom que le rôle. Pour plus d'informations sur les profils d'instance, consultez Utilisation des profils d'instance. Pour de plus amples informations sur le lancement d'une instance EC2 avec un rôle, consultez Contrôle de l'accès aux ressources Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.

Pour créer un profil d'instance et y stocker le rôle (AWS CLI)

1. Créez un profil d'instance : aws iam create-instance-profile
2. Ajoutez le rôle au profil d'instance : aws iam add-role-to-instance-profile

L' AWS CLI exemple de jeu de commandes ci-dessous illustre les deux premières étapes pour créer un rôle et associer des autorisations. Il présente également les deux étapes de la création d'un profil d'instance et de l'ajout du rôle au profil. Cet exemple de politique de confiance permet au service Amazon EC2 d'endosser le rôle et d'afficher le compartiment Amazon S3 `example_bucket`. L'exemple suppose également que vous utilisiez un ordinateur client exécutant Windows et que vous ayez déjà configuré votre interface de ligne de commande avec vos informations d'identification et

votre région. Pour plus d'informations, consultez la section Configuration de l'interface de ligne de AWS commande.

Dans cet exemple, incluez la politique de confiance suivante dans la première commande lors de la création du rôle. Cette politique de confiance permet au service Amazon EC2 d'endosser le rôle.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"Service": "ec2.amazonaws.com"},
    "Action": "sts:AssumeRole"
  }
}
```

Lorsque vous utilisez la deuxième commande, vous devez attacher une politique d'autorisations au rôle. L'exemple de politique d'autorisations suivant permet au rôle d'exécuter uniquement l'action `ListBucket` sur le compartiment Amazon S3 `example_bucket`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Pour créer ce rôle `Test-Role-for-EC2`, vous devez d'abord enregistrer la politique de confiance précédente avec le nom `trustpolicyforec2.json` et la politique d'autorisations précédente avec le nom `permissionspolicyforec2.json` dans le répertoire `policies` de votre disque local `C:`. Vous pouvez ensuite utiliser les commandes suivantes pour créer le rôle, attacher la politique, créer le profil d'instance et ajouter le rôle au profil d'instance.

```
# Create the role and attach the trust policy that allows EC2 to assume this role.
$ aws iam create-role --role-name Test-Role-for-EC2 --assume-role-policy-document
 file://C:\policies\trustpolicyforec2.json

# Embed the permissions policy (in this example an inline policy) to the role to
 specify what it is allowed to do.
$ aws iam put-role-policy --role-name Test-Role-for-EC2 --policy-name Permissions-
Policy-For-Ec2 --policy-document file://C:\policies\permissionspolicyforec2.json

# Create the instance profile required by EC2 to contain the role
$ aws iam create-instance-profile --instance-profile-name EC2-ListBucket-S3

# Finally, add the role to the instance profile
$ aws iam add-role-to-instance-profile --instance-profile-name EC2-ListBucket-S3 --
role-name Test-Role-for-EC2
```

Lorsque vous lancez l'instance EC2, spécifiez le nom du profil d'instance sur la page Configurer les détails de l'instance si vous utilisez la AWS console. Si vous utilisez la commande CLI `aws ec2 run-instances`, précisez le paramètre `--iam-instance-profile`.

Création d'un rôle pour un service (AWS API)

La création d'un rôle à partir de l' AWS API implique plusieurs étapes. Lorsque vous utilisez la console pour créer un rôle, la plupart des étapes sont exécutées automatiquement pour vous, mais avec l'API vous devez exécuter explicitement chaque étape vous-même. Vous devez créer le rôle et lui attribuer une politique d'autorisations. Si le service concerné est Amazon EC2 vous devez également créer un profil d'instance et lui ajouter le rôle. Vous pouvez également définir la limite d'autorisations pour votre rôle.

Pour créer un rôle pour AWS service (AWS API)

1. Créez un rôle : CreateRole

   Vous pouvez spécifier un emplacement de fichier pour la politique d'approbation du rôle.
2. Attachez une politique d'autorisations gérées au rôle : AttachRolePolicy

   or

   Créez une politique d'autorisations en ligne pour le rôle : PutRolePolicy
3. (Facultatif) Ajoutez des attributs personnalisés au rôle en joignant des balises : TagRole

Pour de plus amples informations, veuillez consulter Gestion des balises sur les rôles IAM (AWS CLI or AWS API).

4. (Facultatif) Définissez la limite des autorisations pour le rôle : PutRolePermissionsBoundary

   Une limite d'autorisations contrôle les autorisations maximum dont un rôle peut disposer. Les limites d'autorisations constituent une AWS fonctionnalité avancée.

Si vous comptez utiliser le rôle avec Amazon EC2 ou un autre AWS service utilisant Amazon EC2, vous devez stocker le rôle dans un profil d'instance. Un profil d'instance est un conteneur pour un rôle. Chaque profil d'instance peut contenir un rôle uniquement et cette limite ne peut pas être augmentée. Si vous créez le rôle dans le Console de gestion AWS, le profil d'instance est créé pour vous avec le même nom que le rôle. Pour plus d'informations sur les profils d'instance, consultez Utilisation des profils d'instance. Pour de plus amples informations sur le lancement d'une instance Amazon EC2 avec un rôle, consultez Contrôle de l'accès aux ressources Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.

Pour créer un profil d'instance et y stocker le rôle (AWS API)

1. Créez un profil d'instance : CreateInstanceProfile
2. Ajoutez le rôle au profil de l'instance : AddRoleToInstanceProfile

Créer un rôle lié à un service

Un rôle lié à un service est un type unique de rôle IAM directement lié à un service. AWS Service-linked les rôles sont prédéfinis par le service et incluent toutes les autorisations dont le service a besoin pour appeler d'autres AWS services en votre nom. Le service lié définit aussi la manière dont vous créez, modifiez et supprimez un rôle lié à un service. Un service peut créer ou supprimer automatiquement le rôle. Il peut vous permettre de créer, modifier ou supprimer le rôle dans le cadre des opérations d'un assistant ou d'un processus du service. Ou il peut vous demander d'utiliser IAM pour créer ou supprimer le rôle. Quelle que soit la méthode, les rôles liés à un service simplifient le processus de configuration d'un service étant donné que vous n'avez pas besoin d'ajouter manuellement les autorisations requises pour que le service effectue des actions en votre nom.

> **Note**
>
> N'oubliez pas que les fonctions du service sont différentes des rôles liés à un service. Un rôle de service est un rôle IAM qu'un service endosse pour accomplir des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer un rôle de service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM. Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut assumer le rôle d'effectuer une action en votre nom. Service-linked les rôles apparaissent dans votre compte Compte AWS et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

Le service lié définit les autorisations de ses rôles liés à un service et, sauf définition contraire, seul ce service peut endosser les rôles. Les autorisations définies comprennent la politique de confiance et la politique d'autorisation. De plus, cette politique d'autorisation ne peut pas être attachée à une autre entité IAM.

Avant que vous ne puissiez supprimer les rôles, vous devez d'abord supprimer les ressources qui leur sont associées. Ainsi, vous ne risquez pas de supprimer involontairement l'autorisation d'accéder aux ressources.

> **Tip**
>
> Pour plus d'informations sur les services qui prennent en charge l'utilisation de rôles liés à des services, consultez AWS services qui fonctionnent avec IAM et recherchez les services pour lesquels la mention Oui apparaît dans la colonne Service-Linked Rôle. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Service-linked autorisations de rôle

Vous devez configurer les autorisations d'une entité IAM (utilisateur ou rôle) de manière à permettre à l'utilisateur ou au rôle de créer ou modifier le rôle lié à un service.

> **Note**
>
> L'ARN d'un rôle lié à un service comprend un principal de service indiqué dans les stratégies ci-dessous comme *SERVICE-NAME*.amazonaws.com. N'essayez pas de deviner le principal

du service, car il distingue les majuscules et minuscules et le format peut varier d'un AWS service à l'autre. Pour afficher le principal de service d'un service, consultez la documentation de son rôle lié à un service.

Pour permettre à une entité IAM de créer un rôle spécifique lié à un service

Ajoutez la politique suivante à l'entité IAM qui doit créer le rôle lié à un service.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*",
            "Condition": {"StringLike": {"iam:AWSServiceName": "SERVICE-
NAME.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
        }
    ]
}
```

Pour permettre à une entité IAM de créer un rôle lié à un service

Ajoutez l'instruction suivante à la politique d'autorisation de l'entité IAM qui doit créer un rôle lié à un service, ou un rôle de service incluant les politiques requises. Cette instruction de politique n'autorise pas l'entité IAM à attacher une politique à ce rôle.

```
{
    "Effect": "Allow",
    "Action": "iam:CreateServiceLinkedRole",
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Pour permettre à une entité IAM de modifier la description de rôles liés à un service

Ajoutez l'instruction suivante à la politique d'autorisation de l'entité IAM qui doit modifier la description d'un rôle lié à un service ou d'un rôle de service.

```
{
    "Effect": "Allow",
    "Action": "iam:UpdateRoleDescription",
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Pour permettre à une entité IAM de supprimer un rôle spécifique lié à un service

Ajoutez l'instruction suivante à la politique d'autorisation de l'entité IAM qui doit supprimer le rôle lié à un service.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
}
```

Pour permettre à une entité IAM de supprimer un rôle lié à un service

Ajoutez l'instruction suivante à la politique d'autorisation de l'entité IAM qui doit supprimer un rôle lié à un service, mais pas le rôle de service.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
```

```
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Pour permettre à une entité IAM de transmettre un rôle existant au service

Certains AWS services vous permettent de transmettre un rôle existant au service, au lieu de créer un nouveau rôle lié au service. Pour ce faire, un utilisateur doit disposer des autorisations nécessaires pour transmettre le rôle au service. Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit transmettre un rôle. Cette instruction de politique autorise également l'entité à afficher une liste des rôles à partir de laquelle ils peuvent choisir le rôle à transmettre. Pour de plus amples informations, veuillez consulter Accorder à un utilisateur l'autorisation de transmettre un rôle à un AWS web.

```
{
  "Sid": "PolicyStatementToAllowUserToListRoles",
  "Effect": "Allow",
  "Action": ["iam:ListRoles"],
  "Resource": "*"
},
{
  "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
  "Effect": "Allow",
  "Action": [ "iam:PassRole" ],
  "Resource": "arn:aws:iam::account-id:role/my-role-for-XYZ"
}
```

Autorisations indirectes avec rôles liés à un service

Les autorisations accordées par un rôle lié à un service peuvent être transférées indirectement à d'autres utilisateurs et rôles. Lorsqu'un rôle lié à un service est utilisé par un AWS service, ce rôle lié à un service peut utiliser ses propres autorisations pour appeler d'autres services. AWS Cela signifie que les utilisateurs et les rôles ayant l'autorisation d'appeler un service qui utilise un rôle lié à un service peuvent avoir un accès indirect aux services auxquels ce rôle lié à un service peut accéder.

Par exemple, lorsque vous créez une instance de base données Amazon RDS, un rôle lié à un service pour RDS est automatiquement créé s'il n'existe pas déjà. Ce rôle lié à un service permet à RDS d'appeler Amazon EC2, Amazon SNS, Amazon CloudWatch Logs et Amazon Kinesis en votre nom. Si vous autorisez les utilisateurs et les rôles de votre compte à modifier ou à créer des bases

de données RDS, ils pourront peut-être interagir indirectement avec Amazon EC2, Amazon SNS, les CloudWatch journaux Amazon Logs et les ressources Amazon Kinesis en appelant RDS, car RDS utiliserait son rôle lié à un service pour accéder à ces ressources.

Méthodes pour créer un rôle lié à un service

La méthode que vous utilisez pour créer un rôle lié à un service dépend dudit service. Dans certains cas, vous n'avez pas besoin de créer manuellement un rôle lié à un service. Par exemple, lorsque vous terminez une action donnée (comme créer une ressource) dans le service, le service peut créer le rôle lié au service à votre place. Ou, si vous utilisiez un service avant qu'il ne prenne en charge les rôles liés à un service, alors le service peut avoir créé automatiquement le rôle dans votre compte. Pour en savoir plus, veuillez consulter la section Un nouveau rôle est apparu dans mon AWS compte.

Dans d'autres cas, le service peut prendre en charge la création manuelle d'un rôle lié à un service à l'aide la console, l'API ou de la CLI. Pour plus d'informations sur les services qui prennent en charge l'utilisation de rôles liés à des services, consultez AWS services qui fonctionnent avec IAM et recherchez les services pour lesquels la mention Oui apparaît dans la colonne Service-Linked Rôle. Pour savoir si le service prend en charge la création du rôle lié à un service, choisissez le lien Oui pour afficher la documentation du rôle lié à ce service.

Si le service ne prend pas en charge la création du rôle, alors vous pouvez utiliser IAM pour créer le rôle lié à un service.

> **Important**
>
> Service-linked les rôles sont pris en compte dans vos rôles IAM dans une Compte AWS certaine limite, mais si vous avez atteint cette limite, vous pouvez toujours créer des rôles liés à un service dans votre compte. Seuls les rôles liés à un service peuvent dépasser la limite sans conséquence.

Création d'un rôle lié à un service (console)

Avant de créer un rôle lié à un service dans IAM, sachez si le service lié crée automatiquement des rôles liés au service et si vous pouvez créer le rôle depuis la console, l'API ou la CLI du service.

Pour créer un rôle lié à un service (console)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.

2. Dans le panneau de navigation de la console IAM, choisissez Rôles. Puis, choisissez Create role (Créer un rôle).
3. Choisissez le type de fonction du service AWS .
4. Choisissez le cas d'utilisation de votre service. Les cas d'utilisation sont définis par le service pour inclure la politique d'approbation requise par le service. Ensuite, choisissez Suivant.
5. Choisissez une ou plusieurs politiques d'autorisation à attacher au rôle. En fonction du cas d'utilisation sélectionné, le service peut effectuer n'importe laquelle des options suivantes :

   - Définissez les autorisations utilisées par le rôle.
   - Vous permet de choisir parmi un ensemble limité d'autorisations.
   - Vous permet de choisir parmi toutes les autorisations.
   - Vous permet de choisir de ne sélectionner aucune stratégie pour le moment, mais de créer les politiques plus tard et de les attacher au rôle.

   Cochez la case en regard de la politique qui affecte les autorisations que vous voulez octroyer au rôle, puis choisissez Suivant.

   > **Note**
   >
   > Les autorisations que vous spécifiez sont disponibles à toutes les entités qui utilisent le rôle. Par défaut, un rôle ne dispose d'aucune autorisation.

6. Pour Nom du rôle, le degré de la personnalisation du nom du rôle est défini par le service. Si le service définit le nom du rôle, alors cette option n'est pas modifiable. Dans d'autres cas, le service peut définir un préfixe pour le rôle ou vous laisser saisir un suffixe facultatif.

   Si possible, saisissez un suffixe de nom de rôle à ajouter au nom par défaut. Ce suffixe vous aide à identifier l'objectif de ce rôle. Les noms de rôle de votre compte AWS doivent être uniques. Ils ne sont pas distingués au cas par cas. Par exemple, vous ne pouvez pas créer deux rôles nommés **`<service-linked-role-name>_SAMPLE`** et **`<service-linked-role-name>_sample`**. Différentes entités peuvent référencer le rôle et il n'est donc pas possible de modifier son nom après sa création.
7. (Facultatif) Dans le champ Description, modifiez la description du nouveau rôle lié à un service.
8. Vous ne pouvez pas attacher des balises à des rôles liés à un service lors de la création. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Tags pour Gestion des identités et des accès AWS resources.

9. Passez en revue les informations du rôle, puis choisissez Créer un rôle.

Création d'un rôle lié à un service (AWS CLI)

Avant de créer un rôle lié à un service dans IAM, sachez si le service lié crée automatiquement des rôles liés au service et si vous pouvez créer le rôle depuis la CLI du service. Si la CLI du service n'est pas prise en charge, vous pouvez employer les commandes IAM pour créer un rôle lié à un service avec la politique d'approbation et les politiques en ligne dont le service a besoin pour endosser le rôle.

Pour créer un rôle lié à un service (AWS CLI)

Exécutez la commande suivante :

```
aws iam create-service-linked-role --aws-service-name SERVICE-NAME.amazonaws.com
```

Création d'un rôle lié à un service (AWS API)

Avant de créer un rôle lié à un service dans IAM, sachez si le service lié crée automatiquement des rôles liés au service et si vous pouvez créer le rôle depuis l'API du service. Si l'API de service n'est pas prise en charge, vous pouvez utiliser l' AWS API pour créer un rôle lié à un service avec la politique de confiance et les politiques intégrées dont le service a besoin pour assumer le rôle.

Pour créer un rôle lié à un service (API)AWS

Utilisez l'appel d'API CreateServiceLinkedRole. Dans la demande, spécifiez un nom de service sous la forme *SERVICE_NAME_URL*`.amazonaws.com`.

Par exemple, pour créer le rôle lié à un service Robots Lex, utilisez `lex.amazonaws.com`.

Créer un rôle pour un fournisseur d'identité tiers

Vous pouvez utiliser des fournisseurs d'identité au lieu de créer des utilisateurs IAM dans votre Compte AWS. Avec un fournisseur d'identité (IdP), vous pouvez gérer vos identités d'utilisateurs en dehors de l'extérieur AWS et leur donner les autorisations d'accéder aux AWS ressources de votre compte. Pour plus d'informations sur la fédération et les fournisseurs d'identité, consultez Fournisseurs d'identité et fédération au sein AWS.

Création d'un rôle pour les principaux fédérés OIDC et SAML (console)

Les procédures permettant de créer un rôle dépendent des fournisseurs tiers choisis :

- Pour OpenID Connect (OIDC), consultez Création d'un rôle pour la fédération OpenID Connect (console).
- Pour SAML 2.0, consultez Création d'un rôle pour la fédération SAML 2.0 (console).

Création d'un rôle pour l'accès fédéré (AWS CLI)

Les étapes à suivre pour la création d'un rôle pour les fournisseurs d'identité pris en charge (OIDC ou SAML) à partir de la AWS CLI sont identiques. La différence porte sur le contenu de la politique d'approbation que vous créez dans les étapes préalables requises. Commencez par suivre les étapes de la section Prérequis pour le type de fournisseur que vous utilisez :

- Pour un fournisseur OIDC, consultez Prérequis pour la création d'un rôle pour OIDC.
- Pour un fournisseur SAML, consultez Prérequis pour la création d'un rôle pour SAML.

La création d'un rôle à partir de AWS CLI implique plusieurs étapes. Lorsque vous utilisez la console pour créer un rôle, la plupart des étapes sont effectuées pour vous, mais AWS CLI vous devez effectuer chaque étape vous-même de manière explicite. Vous devez créer le rôle et lui attribuer une politique d'autorisations. Vous pouvez également définir la limite d'autorisations pour votre rôle.

Pour créer un rôle (AWS CLI)

1. Créez un rôle : aws iam create-role
2. Associez une politique d'autorisation au rôle : aws iam attach-role-policy

   or

   Créez une politique d'autorisation intégrée pour le rôle : aws iam put-role-policy
3. (Facultatif) Ajoutez des attributs personnalisés au rôle en associant des balises : aws iam tag-role

   Pour de plus amples informations, veuillez consulter Gestion des balises sur les rôles IAM (AWS CLI or AWS API).
4. (Facultatif) Définissez la limite des autorisations pour le rôle : aws iam put-role-permissions-boundary

   Une limite d'autorisations contrôle les autorisations maximum dont un rôle peut disposer. Les limites d'autorisations constituent une AWS fonctionnalité avancée.

L'exemple suivant illustre les deux premières étapes les plus courantes pour créer un rôle de fournisseur d'identité dans un environnement simple. Cet exemple permet à tout utilisateur du compte `123456789012` d'endosser le rôle et d'afficher le compartiment Amazon S3 `example_bucket`. Cet exemple suppose également que vous exécutez le AWS CLI sur un ordinateur exécutant Windows et que vous l'avez déjà configuré AWS CLI avec vos informations d'identification. Pour plus d'informations, consultez Configuration de l' AWS Command Line Interface.

L'exemple de politique de confiance suivant est conçu pour une application mobile si l'utilisateur se connecte à l'aide d'Amazon Cognito. Dans cet exemple, *`us-east:12345678-ffff-ffff-ffff-123456`* représente l'ID du pool d'identités attribué par Amazon Cognito.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Sid": "RoleForCognito",
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east:12345678-ffff-ffff-ffff-123456"}}
    }
}
```

La politique d'autorisations suivante permet à toute personne endossant le rôle d'exécuter uniquement l'action `ListBucket` sur le compartiment Amazon S3 `example_bucket`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Pour créer ce rôle `Test-Cognito-Role`, vous devez d'abord enregistrer la politique de confiance précédente avec le nom `trustpolicyforcognitofederation.json` et la politique d'autorisations précédente avec le nom `permspolicyforcognitofederation.json` dans le dossier `policies` de votre disque local `C:`. Vous pouvez ensuite utiliser les commandes suivantes pour créer le rôle et attacher la politique en ligne.

```
# Create the role and attach the trust policy that enables users in an account to
 assume the role.
$ aws iam create-role --role-name Test-Cognito-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforcognitofederation.json

# Attach the permissions policy to the role to specify what it is allowed to do.
aws iam put-role-policy --role-name Test-Cognito-Role --policy-name
 Perms-Policy-For-CognitoFederation --policy-document file://C:\policies
\permspolicyforcognitofederation.json
```

Création d'un rôle pour l'accès fédéré (AWS API)

Les étapes à suivre pour la création d'un rôle pour les fournisseurs d'identité pris en charge (OIDC ou SAML) à partir de la AWS CLI sont identiques. La différence porte sur le contenu de la politique d'approbation que vous créez dans les étapes préalables requises. Commencez par suivre les étapes de la section Prérequis pour le type de fournisseur que vous utilisez :

- Pour un fournisseur OIDC, consultez Prérequis pour la création d'un rôle pour OIDC.
- Pour un fournisseur SAML, consultez Prérequis pour la création d'un rôle pour SAML.

Pour créer un rôle (AWS API)

1. Créez un rôle : CreateRole
2. Associez une politique d'autorisation au rôle : AttachRolePolicy

   or

   Créez une politique d'autorisation intégrée pour le rôle : PutRolePolicy
3. (Facultatif) Ajoutez des attributs personnalisés au rôle en attachant des balises : TagRole

   Pour de plus amples informations, veuillez consulter Gestion des balises sur les rôles IAM (AWS CLI or AWS API).
4. (Facultatif) Définissez la limite des autorisations pour le rôle : PutRolePermissionsBoundary

Une limite d'autorisations contrôle les autorisations maximum dont un rôle peut disposer. Les limites d'autorisations constituent une AWS fonctionnalité avancée.

Création d'un rôle pour la fédération OpenID Connect (console)

Vous pouvez utiliser les fournisseurs d'identité fédérés OpenID Connect (OIDC) au lieu de créer des Gestion des identités et des accès AWS utilisateurs dans votre. Compte AWS Avec un fournisseur d'identité (IdP), vous pouvez gérer les identités de vos utilisateurs en dehors de AWS celui-ci et leur accorder les autorisations nécessaires pour accéder aux AWS ressources de votre compte. Pour plus d'informations sur la fédération et IdPs, consultezFournisseurs d'identité et fédération au sein AWS.

Prérequis pour la création d'un rôle pour OIDC

Avant de pouvoir créer un rôle pour la fédération OIDC, vous devez tout d'abord suivre les étapes requises suivantes.

Pour préparer la création d'un rôle pour la fédération OIDC

1. Inscrivez-vous auprès d'un ou plusieurs services offrant une identité OIDC fédérée. Si vous créez une application qui a besoin d'accéder à vos AWS ressources, vous devez également configurer votre application avec les informations du fournisseur. Ainsi, le fournisseur vous communique un ID d'application ou de public unique pour votre application. (Les fournisseurs utilisent une terminologie différente pour ce processus. Ce guide utilise le terme configurer pour désigner le processus d'identification de votre application auprès du fournisseur.) Vous pouvez configurer plusieurs applications auprès de chaque fournisseur, ou plusieurs fournisseurs avec une seule application. Consultez les informations sur l'utilisation des fournisseurs d'identité comme suit :
   - Login with Amazon Developer Center
   - Add Facebook Login to Your App or Website sur le site des développeurs Facebook.
   - Using OAuth 2.0 for Login (OpenID Connect) sur le site des développeurs Google.
2. Après avoir reçu les informations requises de la part de l'IdP, créez un IdP dans IAM. Pour de plus amples informations, veuillez consulter Création d'un fournisseur d'identité OpenID Connect (OIDC) dans IAM.

> **Important**
>
> Si vous utilisez un fournisseur d'identités OIDC de Google, Facebook ou Amazon Cognito, ne créez pas d'IdP IAM distinct dans l' Console de gestion AWS. Ces fournisseurs d'identité OIDC sont déjà intégrés AWS et vous pouvez les utiliser. Ignorez cette étape et créez de nouveaux rôles à l'aide de votre IdP à l'étape suivante.

3. Préparez les politiques relatives au rôle que les IdP-authenticated utilisateurs vont assumer. Comme n'importe quel rôle, celui d'une application mobile inclut deux politiques. L'une est la politique de confiance qui spécifie la personne capable d'endosser le rôle. L'autre est la politique d'autorisation qui spécifie les actions et les ressources AWS auxquelles l'application mobile peut accéder ou non.

   Pour le Web IdPs, nous vous recommandons d'utiliser Amazon Cognito pour gérer les identités. Dans ce cas, utilisez une politique de confiance semblable à cet exemple.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {
            "StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east-2:12345678-abcd-abcd-abcd-123456"},
            "ForAnyValue:StringLike": {"cognito-identity.amazonaws.com:amr":
 "unauthenticated"}
        }
    }
}
```

   Remplacez `us-east-2:12345678-abcd-abcd-abcd-123456` par l'ID du groupe d'identités qu'Amazon Cognito vous assigne.

Si vous configurez manuellement un fournisseur d'identité OIDC, lorsque vous créez la politique de confiance, vous devez utiliser trois valeurs pour garantir que seule votre application peut endosser le rôle :

- Pour l'élément `Action`, utilisez l'action `sts:AssumeRoleWithWebIdentity`.
- Pour l'élément `Principal`, utilisez la chaîne `{"Federated":`*`providerUrl/providerArn`*`}`.
  - Pour certains OIDC courants IdPs, il *`providerUrl`* s'agit d'une URL. Les exemples suivants incluent des méthodes permettant de spécifier le principal pour certaines applications courantes IdPs :

    ```
    "Principal":{"Federated":"cognito-identity.amazonaws.com"}
    ```

    ```
    "Principal":{"Federated":"www.amazon.com"}
    ```

    ```
    "Principal":{"Federated":"graph.facebook.com"}
    ```

    ```
    "Principal":{"Federated":"accounts.google.com"}
    ```

  - Pour les autres fournisseurs OIDC, utilisez l'Amazon Resource Name (ARN) du fournisseur d'identité OIDC créé dans Step 2, comme dans l'exemple suivant :

    ```
    "Principal":{"Federated":"arn:aws:iam::123456789012:oidc-provider/
    server.example.com"}
    ```

- Pour l'élément `Condition`, utilisez une condition `StringEquals` pour limiter les autorisations. Testez l'ID du groupe d'identités pour Amazon Cognito ou l'ID d'application pour les autres fournisseurs. L'ID du groupe d'identité doit correspondre à l'ID d'application obtenu lors de la configuration de l'application avec le fournisseur d'identité. Cette correspondance entre les ID permet de vérifier que la demande provient bien de votre application.

> **Note**
>
> Les rôles IAM pour les groupes d'identités Amazon Cognito font confiance au principal de service `cognito-identity.amazonaws.com` pour assumer ce rôle. Les rôles de ce type doivent contenir au moins une clé de condition pour limiter les principaux qui peuvent assumer le rôle.
> Des considérations supplémentaires s'appliquent aux groupes d'identités Amazon Cognito qui assument des rôles IAM entre comptes. Les politiques d'approbation

associées à ces rôles doivent accepter le principal de service `cognito-identity.amazonaws.com` et contenir la clé de condition `aud` permettant de limiter l'attribution de rôles aux utilisateurs issus des groupes d'identités que vous souhaitez. Une politique qui fait confiance aux groupes d'identités Amazon Cognito sans cette condition crée un risque qu'un utilisateur d'un groupe d'identités non prévu puisse assumer le rôle. Pour plus d'informations, consultez la section Politiques d'approbation pour les rôles IAM dans l'authentification de base (classique) dans le Guide du développeur Amazon Cognito.

Créez un élément de condition semblable à un des exemples suivants, en fonction du fournisseur d'identité que vous utilisez :

```
"Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud":
"us-east:12345678-ffff-ffff-ffff-123456"}}
```

```
"Condition": {"StringEquals": {"www.amazon.com:app_id":
"amzn1.application-oa2-123456"}}
```

```
"Condition": {"StringEquals": {"graph.facebook.com:app_id":
"111222333444555"}}
```

```
"Condition": {"StringEquals": {"accounts.google.com:aud":
"66677788899900pro0"}}
```

Pour les fournisseurs OIDC, utilisez l'URL complète du fournisseur d'identité OIDC avec la clé de contexte `aud`, comme dans exemple suivant :

```
"Condition": {"StringEquals": {"server.example.com:aud":
"appid_from_oidc_idp"}}
```

> **Note**
>
> Les valeurs du principal dans la politique de confiance pour le rôle sont propres à un fournisseur d'identité. Un rôle pour OIDC ne peut spécifier qu'un seul principal. Par conséquent, si l'application mobile autorise des utilisateurs à se connecter à partir de plusieurs fournisseurs d'identité, créez un rôle distinct pour chaque fournisseur d'identité

que vous souhaitez prendre en charge. Créez des politiques de confiance distinctes pour chaque fournisseur d'identité.

Si un utilisateur utilise une application mobile pour se connecter à partir de Login with Amazon (Connexion avec Amazon, l'exemple suivant de politique de confiance s'appliquerait. Dans l'exemple, *amzn1.application-oa2-123456* représente l'ID d'application attribué par Amazon lorsque vous avez configuré l'application à l'aide de Login with Amazon.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForLoginWithAmazon",
          "Effect": "Allow",
          "Principal": {"Federated": "www.amazon.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"www.amazon.com:app_id":
 "amzn1.application-oa2-123456"}}
      }]
  }
```

Si un utilisateur utilise une application mobile pour se connecter à partir de Facebook, l'exemple suivant de politique de confiance s'appliquerait. Dans cet exemple, *111222333444555* représente l'identifiant d'application attribué par Facebook.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForFacebook",
          "Effect": "Allow",
          "Principal": {"Federated": "graph.facebook.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
```

```
          "Condition": {"StringEquals": {"graph.facebook.com:app_id":
 "111222333444555"}}
      }]
  }
```

Si un utilisateur utilise une application mobile pour se connecter à partir de Google, l'exemple suivant de politique de confiance s'appliquerait. Dans cet exemple, *666777888999000* représente l'ID d'application attribué par Google.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForGoogle",
          "Effect": "Allow",
          "Principal": {"Federated": "accounts.google.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"accounts.google.com:aud":
 "666777888999000"}}
      }]
  }
```

Si un utilisateur utilise une application mobile pour se connecter depuis Amazon Cognito, l'exemple de politique de confiance suivant s'applique. Dans cet exemple, *us-east:12345678-ffff-ffff-ffff-123456* représente l'ID du pool d'identités attribué par Amazon Cognito.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForCognito",
          "Effect": "Allow",
          "Principal": {"Federated": "cognito-identity.amazonaws.com"},
```

```
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"cognito-
identity.amazonaws.com:aud": "us-east:12345678-ffff-ffff-ffff-123456"}}
      }]
  }
```

Création d'un rôle pour OIDC

Après avoir exécuté les prérequis, vous pouvez créer le rôle dans IAM. Pour les fournisseurs d'identité OpenID Connect (OIDC) partagés reconnus (IdPs), IAM exige une évaluation explicite des revendications spécifiques dans les jetons Web JSON (JWT), appelés contrôles des fournisseurs d'identité. Pour plus d'informations sur les OIDC IdPs dotés de contrôles de fournisseur d'identité, consultez. Identity-provider contrôles pour les fournisseurs OIDC partagés

La procédure suivante décrit comment créer une le rôle pour la fédération OIDC dans la Console de gestion AWS. Pour créer un rôle à partir de l' AWS API AWS CLI or, consultez les procédures à l'adresseCréer un rôle pour un fournisseur d'identité tiers.

> **Important**
>
> Si vous utilisez Amazon Cognito, utilisez la console Amazon Cognito pour configurer les rôles. Sinon, utilisez la console IAM pour créer un rôle pour la fédération OIDC.

Pour créer un rôle IAM pour la fédération OIDC

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation, choisissez Roles (Rôles), puis Create role (Créer un rôle).
3. Choisissez l'identité Web comme type d'entité d'approbation et sélectionnez Suivant.
4. Pour Identity provider (Fournisseur d'identité), choisissez l'IdP de votre rôle :
   - Si vous créez un rôle pour un fournisseur d'identité Web individuel, choisissez entre Login with Amazon (Connexion avec Amazon), Facebook ou Google.

> **Note**
>
> Vous devez créer un rôle distinct pour chaque IdP que vous souhaitez prendre en charge.

- Si vous souhaitez créer un rôle de scénario avancé pour Amazon Cognito, choisissez Amazon Cognito.

> **Note**
>
> Vous ne devez créer manuellement un rôle à utiliser avec Amazon Cognito que lorsque vous travaillez sur un scénario avancé. Sinon, Amazon Cognito peut créer des rôles pour vous. Pour plus d'informations sur Amazon Cognito, consultez Fournisseurs d'identité externes - Pools d'identités (identités fédérées) dans le Manuel du développeur Amazon Cognito.

- Si vous souhaitez créer un rôle pour les GitHub actions, vous devez commencer par ajouter le fournisseur GitHub OIDC à IAM. Après avoir ajouté le fournisseur GitHub OIDC à IAM, choisissez token.actions.githubusercontent.com.

> **Note**
>
> Pour plus d'informations sur la façon de configurer le fournisseur OIDC GitHub de manière AWS à ce qu'il fasse confiance en tant qu'identité fédérée, consultez la GitHub documentation - Configuration d'OpenID Connect dans Amazon Web Services. Pour plus d'informations sur les meilleures pratiques pour limiter l'accès aux rôles associés à l'IdP IAM pour GitHub, consultez cette Configuration d'un rôle pour le fournisseur d'identité GitHub OIDC page.

- Si vous souhaitez créer un rôle pour HashiCorp Cloud Platform (HCP) Terraform, vous devez commencer par ajouter le fournisseur OIDC Terraform à IAM. Après avoir ajouté le fournisseur Terraform OIDC à IAM, choisissez app.terraform.io.

> **Important**
>
> Rôles IAM pour HashiCorp Cloud Platform (HCP) Le fournisseur Terraform OIDC doit évaluer la clé de condition IAM dans la politique de confiance des

`app.terraform.io:sub` rôles. Cette clé de condition limite les organisations, les projets, les espaces de travail ou les phases d’exécution de HCP Terraform qui peuvent assumer le rôle. Sans cette condition, votre politique de confiance autorise l'accès à votre rôle et à vos AWS ressources par des identités extérieures à votre organisation, ce qui n'est pas conforme au principe du moindre privilège.
Si vous définissez ou modifiez une politique de confiance pour un rôle associé au fournisseur OIDC HCP Terraform dans votre AWS compte, mais que vous n'évaluez pas la clé de condition IAM`app.terraform.io:sub`, vous recevrez un message d'erreur. De plus, AWS STS refusera les requêtes d’autorisation si votre politique d’approbation des rôles n’évalue pas cette clé de condition.

5. Les informations demandées varient en fonction du fournisseur OIDC que vous choisissez.

   - Saisissez l'identifiant de votre application. L'étiquette de l'identifiant varie selon le fournisseur que vous choisissez :
     - Si vous souhaitez créer un rôle pour Login with Amazon (Connexion avec Amazon), saisissez l'ID d'application dans le champ Application ID (ID d'application).
     - Si vous souhaitez créer un rôle pour Facebook, saisissez l'ID d'application dans le champ Application ID (ID d'application).
     - Si vous souhaitez créer un rôle pour Google, saisissez le nom du public cible dans le champ Audience (Public cible).
     - Si vous souhaitez créer un rôle pour Amazon Cognito, saisissez l'ID du groupe d'identités que vous avez créé pour vos applications Amazon Cognito dans le champ Identity Pool ID (ID du pool d'identités).
   - Si vous souhaitez créer un rôle pour les GitHub actions, entrez les informations suivantes :
     - Pour Audience, choisissez `sts.amazonaws.com`.
     - Pour GitHub l'organisation, entrez le nom de GitHub l'organisation. Le nom de GitHub l'organisation est obligatoire et doit être alphanumérique, y compris des tirets (-). Vous ne pouvez pas utiliser de caractères génériques (* et ?) dans le nom GitHub de l'organisation.
     - (Facultatif) Pour GitHub le référentiel, entrez le nom du GitHub référentiel. Si vous ne précisez pas de valeur, la valeur par défaut devient un caractère de remplacement (*).
     - (Facultatif) Pour GitHub la succursale, entrez le nom de la GitHub succursale. Si vous ne précisez pas de valeur, la valeur par défaut devient un caractère de remplacement (*).
   - Si vous souhaitez créer un rôle pour HashiCorp Cloud Platform (HCP) Terraform, entrez les informations suivantes :

- Pour Audience, choisissez `aws.workload.identity`.
- Pour Organisation, saisissez le nom de l'organisation. Vous pouvez spécifier un caractère générique (*) pour toutes les organisations.
- Pour Projet, saisissez le nom du projet. Vous pouvez spécifier un caractère générique (*) pour tous les projets.
- Pour Espace de travail, saisissez le nom de l'espace de travail. Vous pouvez spécifier un caractère générique (*) pour tous les espaces de travail.
- Pour Phase d'exécution, saisissez le nom de la phase d'exécution. Vous pouvez spécifier un caractère générique (*) pour toutes les phases d'exécution.

6. (Facultatif) Pour les Conditions (facultatif), choisissez Ajouter une condition pour créer des conditions supplémentaires à réunir avant que les utilisateurs de votre application ne puissent utiliser les autorisations que le rôle accorde. Par exemple, vous pouvez ajouter une condition qui autorise l'accès aux AWS ressources uniquement pour un ID utilisateur IAM spécifique. Vous pouvez par ailleurs ajouter des conditions à la politique de confiance après la création du rôle. Pour de plus amples informations, veuillez consulter Mise à jour d'une politique d'approbation de rôle.
7. Vérifiez les informations OIDC que vous avez saisies, puis cliquez sur Suivant.
8. IAM inclut une liste des politiques AWS gérées et gérées par le client dans votre compte. Sélectionnez la politique à utiliser pour la politique d'autorisations ou choisissez Create policy (Créer une politique) pour ouvrir un nouvel onglet de navigateur et créer une nouvelle politique de bout en bout. Pour de plus amples informations, veuillez consulter Création de politiques IAM. Une fois la politique créée, fermez cet onglet et revenez à l'onglet initial. Cochez la case en regard des politiques d'autorisations que vous souhaitez octroyer aux utilisateurs OIDC. Si vous préférez, vous pouvez ne sélectionner aucune stratégie pour le moment, puis les attacher au rôle ultérieurement. Par défaut, un rôle ne dispose d'aucune autorisation.
9. (Facultatif) Définissez une limite d'autorisations. Il s'agit d'une fonctionnalité avancée.

   Ouvrez la section Set permissions boundary (Définir une limite d'autorisations) et choisissez Use a permissions boundary to control the maximum role permissions (Utiliser une limite d'autorisations pour contrôler le nombre maximum d'autorisations de rôle). Sélectionnez la politique à utiliser comme limite d'autorisations.
10. Choisissez Suivant.
11. Pour Role name (Nom du rôle), saisissez un nom de rôle. Les noms de rôles doivent être uniques au sein de votre Compte AWS. Ils ne dépendent pas de la casse. Par exemple, vous ne pouvez pas créer deux rôles nommés **`PRODROLE`** et **`prodrole`**. Étant donné que d'autres AWS

ressources peuvent faire référence au rôle, vous ne pouvez pas modifier le nom du rôle une fois que vous l'avez créé.

12. (Facultatif) Pour Description, saisissez une description pour le nouveau rôle.
13. Pour modifier les cas d'utilisation et les autorisations pour le rôle, choisissez Edit (Modifier) dans les sections Step 1: Select trusted entities (Étape 1 : sélection d'entités de confiance) ou Step 2: Select permissions (Étape 2 : sélection d'autorisations).
14. (Facultatif) Pour ajouter des métadonnées au rôle, attachez des balises en tant que paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Tags pour Gestion des identités et des accès AWS resources.
15. Passez en revue les informations du rôle, puis choisissez Créer un rôle.

Configuration d'un rôle pour le fournisseur d'identité GitHub OIDC

Si vous l'utilisez GitHub en tant que fournisseur d'identité (IdP) OpenID Connect (OIDC), la meilleure pratique consiste à limiter le nombre d'entités pouvant assumer le rôle associé à l'IdP IAM. Lorsque vous incluez une déclaration de condition dans la politique de confiance, vous pouvez limiter le rôle à une GitHub organisation, un référentiel ou une branche spécifique. Vous pouvez utiliser la clé de condition `token.actions.githubusercontent.com:sub` avec des opérateurs de condition de chaîne pour limiter l'accès. Nous vous recommandons de limiter la condition à un ensemble spécifique de référentiels ou de branches au sein de votre GitHub organisation. Pour plus d'informations sur la façon de AWS configurer GitHub l'OIDC comme identité fédérée, consultez la section GitHub Documentation - Configuration d'OpenID Connect dans Amazon Web Services.

Si vous utilisez GitHub des environnements dans des flux de travail ou dans des politiques OIDC, nous vous recommandons vivement d'ajouter des règles de protection à l'environnement pour renforcer la sécurité. Utilisez les branches et les balises de déploiement pour limiter les branches et les balises qui peuvent être déployées dans l'environnement. Pour plus d'informations sur la configuration des environnements avec des règles de protection, consultez la section Branches et balises GitHub de déploiement dans l'article Utilisation des environnements pour le déploiement.

Lorsque GitHub l'IdP OIDC est le principal approuvé pour votre rôle, IAM vérifie la condition de la politique de confiance du rôle pour vérifier que la clé de condition `token.actions.githubusercontent.com:sub` est présente et que sa valeur n'est pas uniquement un caractère générique (* et ?) ou nul. IAM effectue cette vérification lors de la création ou de la mise à jour de la politique IAM. Si la clé de condition `token.actions.githubusercontent.com:sub` est absente ou la valeur clé ne répond pas aux critères de valeur mentionnés, la demande échouera et renverra une erreur.

> **Important**
>
> Si vous ne limitez pas la clé de condition `token.actions.githubusercontent.com:sub` à une organisation ou à un référentiel spécifique, GitHub les actions des organisations ou des référentiels indépendants de votre volonté peuvent assumer les rôles associés à l'IdP GitHub IAM de votre compte. AWS

L'exemple de politique de confiance suivant limite l'accès à l' GitHub organisation, au référentiel et à la succursale définis. La `token.actions.githubusercontent.com:sub` valeur de la clé de condition dans l'exemple suivant est le format de valeur de sujet par défaut documenté par GitHub.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::012345678910:oidc-provider/
token.actions.githubusercontent.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
          "token.actions.githubusercontent.com:sub":
 "repo:GitHubOrg/GitHubRepo:ref:refs/heads/GitHubBranch"
        }
      }
    }
  ]
}
```

L'exemple de condition suivant limite l'accès à l' GitHub organisation et au référentiel définis, mais accorde l'accès à n'importe quelle branche du référentiel.

```
"Condition": {
```

```
  "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
  },
  "StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/GitHubRepo:*"
  }
}
```

L'exemple de condition suivant limite l'accès à n'importe quel référentiel ou branche au sein de l' GitHub organisation définie. Nous vous recommandons de limiter la clé de condition `token.actions.githubusercontent.com:sub` à une valeur spécifique qui limite l'accès aux GitHub actions au sein de votre GitHub organisation.

```
"Condition": {
  "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
  },
  "StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/*"
  }
}
```

Pour plus d'informations sur les clés de fédération OIDC disponibles pour les contrôles de condition dans les politiques, consultez Clés disponibles pour AWS Fédération OIDC.

Création d'un rôle pour la fédération SAML 2.0 (console)

Vous pouvez utiliser la fédération SAML 2.0 au lieu de créer des utilisateurs IAM dans votre. Compte AWS Avec un fournisseur d'identité (IdP), vous pouvez gérer les identités de vos utilisateurs en dehors de AWS celui-ci et leur accorder les autorisations nécessaires pour accéder aux AWS ressources de votre compte. Pour plus d'informations sur la fédération et les fournisseurs d'identité, consultez Fournisseurs d'identité et fédération au sein AWS.

> **Note**
>
> Pour améliorer la résilience de la fédération, nous vous recommandons de configurer votre IdP et votre fédération AWS pour prendre en charge plusieurs points de terminaison de connexion SAML. Pour plus de détails, consultez l'article du blog sur la AWS sécurité Comment utiliser les points de terminaison SAML régionaux pour le basculement.

Prérequis pour la création d'un rôle pour SAML

Avant de pouvoir créer un rôle pour la fédération SAML 2.0, vous devez tout d'abord suivre les étapes requises suivantes.

Pour préparer la création d'un rôle pour la fédération SAML 2.0

1. Avant de créer un rôle pour SAML-based la fédération, vous devez créer un fournisseur SAML dans IAM. Pour de plus amples informations, veuillez consulter Création d'un fournisseur d'identité SAML dans IAM.
2. Préparez les politiques pour le rôle que les utilisateurs authentifiés SAML 2.0 vont endosser. Comme n'importe quel rôle, celui de la fédération SAML inclut deux politiques. L'une est la politique de confiance de rôle qui spécifie la personne capable d'endosser le rôle. L'autre est la politique d'autorisations IAM qui spécifie les AWS actions et les ressources auxquelles l'accès est autorisé ou refusé au principal fédéré SAML.

   Lorsque vous créez la politique de confiance pour votre rôle, vous devez utiliser trois valeurs pour vous assurer que seule votre application peut assumer le rôle :

   - Pour l'élément `Action`, utilisez l'action `sts:AssumeRoleWithSAML`.
   - Pour l'élément `Principal`, utilisez la chaîne `{"Federated":`*`ARNofIdentityProvider`*`}`. Remplacez *`ARNofIdentityProvider`* par l'ARN du fournisseur d'identité SAML que vous avez créé dans Step 1.
   - Pour l'élément `Condition`, utilisez une condition `StringEquals` pour vérifier que l'attribut `saml:aud` de la réponse SAML correspond à l'URL affichée par votre navigateur lorsque vous vous connectez à la console. Cette URL de point de terminaison de connexion correspond à l'attribut du destinataire SAML de votre fournisseur d'identité. Vous pouvez inclure des URL de connexion dans des régions spécifiques. AWS recommande d'utiliser des points de terminaison régionaux au lieu du point de terminaison global pour améliorer la résilience de la fédération. Pour une liste des *`region-code`* valeurs possibles, consultez la colonne Région dans les AWS Sign-In points de .

     Si le chiffrement SAML est obligatoire, l'URL de connexion doit inclure l'identifiant unique qu' AWS attribue à votre fournisseur SAML. Vous pouvez afficher l'identifiant unique en sélectionnant le fournisseur d'identité dans la console IAM afin d'afficher la page de détails.

     `https://`*`region-code`*`.signin.aws.amazon.com/saml/acs/`*`IdP-ID`*

L'exemple suivant de politique de confiance est conçu pour un utilisateur fédéré SAML :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRoleWithSAML",
        "Principal": {
            "Federated": "arn:aws:iam::111122223333:saml-provider/PROVIDER-
NAME"
        },
        "Condition": {
            "StringEquals": {
                "SAML:aud": "https://region-code.signin.aws.amazon.com/saml"
            }
        }
    }
}
```

Remplacez l'ARN principal par l'ARN réel pour le fournisseur SAML que vous avez créé dans IAM. Il utilisera votre propre ID de compte et nom du fournisseur.

Création d'un rôle pour SAML

Après avoir effectué les étapes préalables, vous pouvez créer le rôle de SAML-based fédération.

Pour créer un rôle pour la SAML-based fédération

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
3. Choisissez le type de rôle Fédération SAML 2.0.
4. Pour Select a SAML provider (Sélectionnez un fournisseur SAML), choisissez le fournisseur de votre rôle.

5. Choisissez la méthode de niveau d'accès SAML 2.0.

   - Choisissez Autoriser l'accès par programmation uniquement pour créer un rôle qui peut être assumé par programmation depuis l' AWS API ou. AWS CLI
   - Choisissez Autoriser la programmation et Console de gestion AWS l'accès pour créer un rôle qui peut être assumé par programmation et à partir du. Console de gestion AWS

   Les rôles créés par les deux méthodes sont similaires, mais le rôle qui peut aussi être endossé depuis la console inclut une politique d'approbation contenant une condition particulière. Cette condition vérifie explicitement que l'audience SAML (l'attribut `SAML:aud`) est définie sur le point de terminaison de connexion AWS pour votre fournisseur SAML

6. La procédure de définition des attributs varie en fonction du type d'accès.

   - Si vous créez un rôle pour un accès par programme, choisissez un attribut dans la liste Attribut. Ensuite, dans le champ Value (Valeur), tapez une valeur à inclure dans le rôle. Cette valeur restreint l'accès au rôle aux utilisateurs du fournisseur d'identité dont la réponse d'authentification SAML (assertion) inclut les attributs que vous spécifiez. Vous devez spécifier au moins un attribut afin de garantir la limitation du rôle à un sous-ensemble d'utilisateurs de votre organisation.
   - Si vous créez un rôle pour la programmation et l' Console de gestion AWS accès, la section Sign-in Endpoints définit l'URL que votre navigateur affiche lors de la connexion à la console. Ce point de terminaison est l'attribut de destinataire SAML de votre fournisseur d'identité, qui correspond à la clé de contexte `saml:aud`. Pour de plus amples informations, veuillez consulter Configuration d'assertions SAML pour la réponse d'authentification.

     a. Choisissez des points de terminaison régionaux ou des points de Non-regional terminaison. Nous recommandons d'utiliser plusieurs points de terminaison de connexion SAML régionaux afin d'améliorer la résilience de la fédération.

     b. Pour les régions, choisissez les régions que votre fournisseur SAML prend en charge pour la AWS connexion.

     c. Pour que Sign-in les URL incluent des identifiants uniques, indiquez si les points de terminaison de connexion incluent les identifiants uniques attribués à votre fournisseur d'identité AWS SAML. Cette option est requise pour les assertions SAML cryptées. Pour de plus amples informations, veuillez consulter Fédération SAML 2.0.

7. Pour ajouter d'autres conditions liées aux attributs à la politique d'approbation, choisissez Condition (optional) (Conditions [facultatif]), sélectionnez la condition supplémentaire et spécifiez une valeur.

   > **Note**
   >
   > La liste inclut les attributs SAML les plus couramment utilisés. IAM prend en charge des attributs supplémentaires que vous pouvez utiliser pour créer des conditions. Pour obtenir la liste des attributs pris en charge, veuillez consulter Clés disponibles pour la fédération SAML. Si vous avez besoin d'une condition pour un attribut SAML pris en charge ne figurant pas dans la liste, vous pouvez ajouter cette condition manuellement. Pour ce faire, modifiez la politique de confiance après la création du rôle.

8. Passez en revue les informations d'approbation SAML 2.0, puis choisissez Next: Permissions (Suivant : Autorisations).
9. IAM inclut une liste des politiques AWS gérées et gérées par le client dans votre compte. Sélectionnez la politique à utiliser pour la politique d'autorisations ou choisissez Create policy (Créer une politique) pour ouvrir un nouvel onglet de navigateur et créer une nouvelle politique de bout en bout. Pour de plus amples informations, veuillez consulter Création de politiques IAM. Une fois la politique créée, fermez cet onglet et revenez à l'onglet initial. Cochez la case en regard des politiques d'autorisations que vous souhaitez octroyer aux utilisateurs fédérés SAML. Si vous préférez, vous pouvez ne sélectionner aucune stratégie pour le moment, puis les attacher au rôle ultérieurement. Par défaut, un rôle ne dispose d'aucune autorisation.
10. (Facultatif) Définissez une limite d'autorisations. Il s'agit d'une fonctionnalité avancée.

    Ouvrez la section Set permissions boundary (Définir une limite d'autorisations) et choisissez Use a permissions boundary to control the maximum role permissions (Utiliser une limite d'autorisations pour contrôler le nombre maximum d'autorisations de rôle). Sélectionnez la politique à utiliser comme limite d'autorisations.
11. Choisissez Suivant.
12. Choisissez Suivant : Vérification.
13. Pour Role name (Nom du rôle), saisissez un nom de rôle. Les noms de rôles doivent être uniques au sein de votre Compte AWS. Ils ne sont pas distingués par cas. Par exemple, vous ne pouvez pas créer deux rôles nommés **`PRODROLE`** et **`prodrole`**. Étant donné que d'autres AWS ressources peuvent faire référence au rôle, vous ne pouvez pas modifier le nom du rôle une fois qu'il a été créé.

14. (Facultatif) Pour Description, saisissez une description pour le nouveau rôle.
15. Choisissez Edit (Modifier) dans les sections Step 1: Select trusted entities (Étape 1 : sélection d'entités de confiance) ou Step 2: Add permissions (Étape 2 : ajouter des autorisations) pour modifier les cas d'utilisation et les autorisations pour le rôle.
16. (Facultatif) Ajoutez des métadonnées au rôle en associant les identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Tags pour Gestion des identités et des accès AWS resources.
17. Passez en revue les informations du rôle, puis choisissez Créer un rôle.

Après avoir créé le rôle, vous devez finaliser la relation d'approbation SAML en configurant le logiciel de votre fournisseur d'identité à l'aide d'informations sur AWS. Ces informations incluent les rôles que vous souhaitez que vos utilisateurs fédérés SAML utilisent. Il s'agit de la configuration de la relation d'approbation des parties utilisatrices entre votre fournisseur d'identité et AWS. Pour de plus amples informations, veuillez consulter Configuration de votre IdP SAML 2.0 à l'aide d'une relation d'approbation des parties utilisatrices et ajout de demandes.

Création d'un rôle à l'aide de politiques d'approbation personnalisées

Vous pouvez créer une politique de confiance personnalisée pour déléguer l'accès et permettre à d'autres personnes d'effectuer des actions dans votre Compte AWS. Pour de plus amples informations, veuillez consulter Création de politiques IAM.

Pour plus d'informations sur la façon d'utiliser les rôles pour déléguer des autorisations, consultez Termes et concepts relatifs aux rôles.

Création d'un rôle IAM à l'aide d'une politique d'approbation personnalisée (console)

Vous pouvez utiliser le Console de gestion AWS pour créer un rôle qu'un utilisateur IAM peut assumer. Supposons, par exemple, que votre organisation en dispose Comptes AWS de plusieurs pour isoler un environnement de développement d'un environnement de production. Pour des informations générales sur la création d'un rôle autorisant les utilisateurs du compte de développement à accéder aux ressources du compte de production, consultez Exemple de scénario utilisant des comptes de développement et de production distincts.

Pour créer un rôle à l'aide d'une politique d'approbation personnalisée (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.

2. Dans le panneau de navigation de la console, choisissez Rôles, puis Créer un rôle.
3. Choisissez le type de rôle Custom trust policy (Politique d'approbation personnalisée).
4. Dans la section Custom trust policy (Politique d'approbation personnalisée), saisissez ou collez la politique d'approbation personnalisée pour le rôle. Pour de plus amples informations, veuillez consulter Création de politiques IAM.
5. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Next (Suivant).
6. (Facultatif) Définissez une limite d'autorisations. Il s'agit d'une fonctionnalité avancée disponible pour les fonctions de service, mais pas pour les rôles liés à un service.

   Ouvrez la section Set permissions boundary (Définir une limite d'autorisations) et choisissez Use a permissions boundary to control the maximum role permissions (Utiliser une limite d'autorisations pour contrôler le nombre maximum d'autorisations de rôle). IAM inclut une liste des politiques AWS gérées et gérées par le client dans votre compte. Sélectionnez la politique à utiliser comme limite d'autorisations.
7. Choisissez Suivant.
8. Pour Nom du rôle, le degré de la personnalisation du nom du rôle est défini par le service. Si le service définit le nom du rôle, cette option n'est pas modifiable. Dans d'autres cas, le service peut définir un préfixe pour le rôle ou vous permettre de saisir un suffixe facultatif. Certains services vous permettent de spécifier le nom complet de votre rôle.

   Si possible, saisissez un nom de rôle ou un suffixe de nom de rôle. Les noms de rôles doivent être uniques au sein de votre Compte AWS. Ils ne sont pas distingués au cas par cas. Par exemple, vous ne pouvez pas créer deux rôles nommés **`PRODROLE`** et **`prodrole`**. Dans la mesure AWS où d'autres ressources peuvent faire référence au rôle, vous ne pouvez pas modifier le nom du rôle une fois celui-ci créé.
9. (Facultatif) Pour Description, saisissez une description pour le nouveau rôle.
10. (Facultatif) Choisissez Modifier dans les sections Étape 1 : sélection d'entités de confiance ou Étape 2 : Ajouter des autorisations pour modifier la politique et les autorisations personnalisées pour le rôle.
11. (Facultatif) Ajoutez des métadonnées au rôle en associant les identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Tags pour Gestion des identités et des accès AWS resources.
12. Passez en revue les informations du rôle, puis choisissez Créer un rôle.

Exemples de politiques pour la délégation d'accès

Les exemples suivants montrent comment vous pouvez autoriser ou accorder l' Compte AWS accès aux ressources d'un autre site Compte AWS. Pour apprendre à créer une politique IAM à l'aide de ces exemples de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

Rubriques

- Utiliser des rôles pour déléguer l'accès aux ressources d'un autre Compte AWS resources
- Utilisation d'une politique pour la délégation d'accès à des services
- Utilisation d'une politique basée sur les ressources pour la délégation de l'accès à un compartiment Amazon S3 dans un autre compte
- Utilisation d'une politique basée sur les ressources pour la délégation de l'accès à une file d'attente Amazon SQS dans un autre compte
- Délégation d'accès impossible lorsque l'accès est refusé au compte

Utiliser des rôles pour déléguer l'accès aux ressources d'un autre Compte AWS resources

Pour un didacticiel qui montre comment utiliser les rôles IAM pour accorder aux utilisateurs d'un compte l'accès aux AWS ressources d'un autre compte, consultezTutoriel IAM : accès délégué via AWS comptes utilisant des rôles IAM.

> **Important**
>
> Vous pouvez inclure l'ARN d'un rôle ou d'un utilisateur dans l'élément `Principal` de la politique d'approbation d'un rôle. Lorsque vous enregistrez la politique, AWS transforme l'ARN en un ID principal unique. Cela permet de réduire le risque d'escalade des privilèges par la suppression et la nouvelle création du rôle ou de l'utilisateur. Cet ID n'est pas fréquent dans la console, car il existe également une transformation inverse, pour revenir à l'ARN, lorsque la politique d'approbation est affichée. Cependant, si vous supprimez le rôle ou l'utilisateur, la relation est interrompue. La politique ne s'applique plus, même si vous recréez l'utilisateur ou le rôle, étant donné qu'il ne correspond pas à l'ID du principal stocké dans la politique d'approbation. Dans ce cas, l'ID principal apparaît dans la console car il n'est plus AWS possible de le mapper à un ARN. Le résultat final est que si vous supprimez et recréez un utilisateur ou un rôle référencé dans l'élément `Principal` d'une politique d'approbation,

vous devez modifier le rôle afin de remplacer l'ARN. Il deviendra le nouvel ID du principal lorsque vous enregistrez la politique.

Utilisation d'une politique pour la délégation d'accès à des services

L'exemple suivant illustre une politique qui est peut être attachée à un rôle. La politique permet à deux services, Amazon EMR et Amazon AWS Data Pipeline, d'assumer ce rôle. Les services peuvent ensuite effectuer toutes les tâches autorisées par la politique d'autorisation affectée au rôle (non illustré). Pour définir plusieurs principaux de service, vous ne spécifiez pas deux éléments `Service` ; vous ne devez en avoir qu'un seul. À la place, vous utilisez un tableau de plusieurs principaux de service comme valeur d'un élément `Service` unique.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "elasticmapreduce.amazonaws.com",
          "datapipeline.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Utilisation d'une politique basée sur les ressources pour la délégation de l'accès à un compartiment Amazon S3 dans un autre compte

Dans cet exemple, le compte A utilise une politique basée sur les ressources (une politique de compartiment Amazon S3) pour octroyer au compte B l'accès complet au compartiment S3 du compte A. Ensuite, le compte B crée une politique d'utilisateur IAM pour déléguer l'accès au compartiment du compte A à l'un des utilisateurs du compte B.

La politique de compartiment S3 du compte A peut se présenter comme suit. Dans cet exemple, le compartiment S3 du compte A est nommé amzn-s3-demo-bucket, et le numéro de compte du compte B est 111122223333. Il ne spécifie pas d'utilisateurs ou de groupes dans le compte B, simplement le compte proprement dit.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "AccountBAccess1",
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket",
      "arn:aws:s3:::amzn-s3-demo-bucket/*"
    ]
  }
}
```

En variante, le compte A peut utiliser des Listes de contrôle d'accès (ACL) Amazon S3 pour accorder l'accès au compte B à un compartiment S3 ou à un seul objet dans un compartiment. Dans ce cas, seule change la façon dont le compte A accorde l'accès au compte B. Le compte B utilise toujours une politique pour déléguer l'accès à un groupe IAM dans le compte B, comme décrit dans la dernière partie de cet exemple. Pour plus d'informations sur le contrôle de l'accès aux compartiments et aux objets S3, consultez la section Contrôle d'accès dans le guide de l'utilisateur d'Amazon Simple Storage Service.

L'administrateur du compte B peut créer l'exemple de politique suivant. La politique donne un accès en lecture à un groupe ou à un utilisateur au sein du compte B. La politique précédente accorde l'accès au compte B. Toutefois, des groupes et des utilisateurs spécifiques au sein du compte B ne peuvent pas accéder à la ressource tant qu'une politique de groupe ou d'utilisateur ne leur accorde pas explicitement des autorisations pour cette ressource. Les autorisations de cette politique peuvent uniquement être un sous-ensemble de celles de la politique intercompte précédente. Le compte B ne peut pas accorder plus d'autorisations à ses utilisateurs et groupes que le compte A ne lui a accordé dans la première politique. Dans cette politique, l'élément `Action` est défini explicitement pour

autoriser uniquement les actions `List`, tandis que l'élément `Resource` de cette politique correspond à l'élément `Resource` pour la politique de compartiment implémentée par le compte A.

Pour implémenter cette politique, le compte B utilise IAM pour l'attacher à l'utilisateur (ou au groupe) approprié dans le compte B.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:List*",
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket",
      "arn:aws:s3:::amzn-s3-demo-bucket/*"
    ]
  }
}
```

Utilisation d'une politique basée sur les ressources pour la délégation de l'accès à une file d'attente Amazon SQS dans un autre compte

Dans l'exemple suivant, le compte A est doté d'une file d'attente Amazon SQS qui utilise une politique basée sur les ressources attachée à celle-ci pour accorder au compte B l'accès à la file d'attente. Le compte B utilise ensuite une politique de groupe IAM pour déléguer l'accès à un groupe du compte B.

L'exemple de politique de file d'attente suivant accorde au compte B l'autorisation d'effectuer les actions `SendMessage` et `ReceiveMessage` dans la file d'attente du compte A nommée queue1, mais uniquement entre midi et 15 heures le 30 novembre 2014. Le numéro de compte du compte B est 1111-2222-3333. Le compte A utilise Amazon SQS pour implémenter cette politique.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
```

```
    "Action": [
      "sqs:SendMessage",
      "sqs:ReceiveMessage"
    ],
    "Resource": ["arn:aws:sqs:*:123456789012:queue1"],
    "Condition": {
      "DateGreaterThan": {"aws:CurrentTime": "2014-11-30T12:00Z"},
      "DateLessThan": {"aws:CurrentTime": "2014-11-30T15:00Z"}
    }
  }
}
```

La politique du compte B pour la délégation d'accès à un groupe dans le compte B peut se présenter comme l'exemple suivant. Le compte B utilise IAM pour attacher cette politique à un groupe (ou utilisateur).

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sqs:*",
    "Resource": "arn:aws:sqs:*:123456789012:queue1"
  }
}
```

Dans l'exemple de politique utilisateur IAM précédent, le compte B utilise un caractère générique pour accorder à son utilisateur l'accès à toutes les actions Amazon SQS dans la file d'attente du compte A. Cependant, le compte B peut déléguer l'accès dans la mesure où celui-ci lui a été accordé. Le groupe du compte B qui a une seconde politique peut accéder à la file d'attente uniquement entre midi et 15 heures le 30 novembre 2014. L'utilisateur peut uniquement effectuer les actions `SendMessage` et `ReceiveMessage`, tel que défini dans la politique de file d'attente Amazon SQS du compte A.

Délégation d'accès impossible lorsque l'accès est refusé au compte

An Compte AWS ne peut pas déléguer l'accès aux ressources d'un autre compte si l'autre compte a explicitement refusé l'accès au compte parent de l'utilisateur. Le rejet se propage aux utilisateurs sous ce compte, qu'ils aient ou non des politiques existantes leur accordant l'accès.

Par exemple, le compte A crée une politique de compartiment pour son compartiment S3 qui refuse explicitement au compte B l'accès à ce compartiment. Mais le compte B crée une politique d'utilisateur IAM qui accorde à un utilisateur du compte B l'accès au compartiment du compte A. Le refus explicite appliqué au compartiment S3 du compte A est propagé aux utilisateurs du compte B. Il remplace la politique d'utilisateur IAM qui accorde l'accès aux utilisateurs du compte B. (Pour des informations détaillées sur la façon dont les autorisations sont évaluées, consultez Logique d'évaluation de politiques.)

La politique de compartiment du compte A peut se présenter comme suit. Dans cet exemple, le compartiment S3 du compte A est nommé amzn-s3-demo-bucket, et le numéro de compte du compte B est 1111-2222-3333. Le compte A utilise Amazon S3 pour implémenter cette politique.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "AccountBDeny",
    "Effect": "Deny",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
  }
}
```

Ce refus explicite remplace toutes les politiques au sein du compte B qui accordent l'accès au compartiment S3 du compte A.

Créez des rôles automatiquement avec le gestionnaire de rôles

De nombreux AWS services dépendent des rôles IAM pour effectuer des actions en votre nom. Le gestionnaire de rôles IAM est un paramètre de compte facultatif qui provisionne automatiquement ces rôles, de sorte que vous n'avez pas à les configurer vous-même. Le gestionnaire de rôles est

conçu pour vous aider à commencer à créer avec AWS des services sans configuration manuelle supplémentaire d'IAM.

Le gestionnaire de rôles est disponible dans certaines consoles de AWS service répertoriées the section called "AWS consoles de service prenant en charge le gestionnaire de rôles" sur cette page.

Comment fonctionne le gestionnaire de rôles

Lorsque le gestionnaire de rôles est activé, il fournit des rôles adaptés au service et à la ressource dans le cadre du flux de travail de console correspondant. Les consoles de service prises en charge indiqueront à quel moment le gestionnaire de rôles est disponible. Aucune action supplémentaire n'est requise pour accepter le rôle fourni par le gestionnaire de rôles. Par exemple, lorsque vous créez une fonction AWS Lambda dans la console, Lambda utilise le gestionnaire de rôles pour créer le rôle d'exécution de la fonction par défaut. Pour utiliser un autre rôle, choisissez un rôle existant ou créez-en un nouveau manuellement.

Le gestionnaire de rôles fonctionne en appliquant un modèle de rôle, une définition de la politique de confiance d'un rôle et des autorisations pour un service et un cas d'utilisation. Pour en savoir plus sur le modèle utilisé par chaque service, consultezVue d'ensemble des modèles de rôles. Un modèle pour une tâche définie octroie uniquement les actions dont la tâche a besoin. Un modèle pour le travail ouvert, tel que l'exécution de votre propre code, accorde un accès étendu aux AWS API afin que vous puissiez créer sans accorder l'accès à chaque service vous-même.

Les rôles IAM créés par le gestionnaire de rôles se comportent comme n'importe quel autre rôle que vous créez. Vous pouvez les consulter, les modifier ou les supprimer. Chaque rôle enregistre également le modèle de rôle dont il provient, afin que vous puissiez identifier les rôles créés par le responsable des rôles. AWS CloudTrail enregistre lorsque le gestionnaire de rôles crée des rôles afin que vous puissiez vérifier quand et comment chaque rôle a été créé. La création du rôle de gestionnaire de rôles apparaît CloudTrail sous forme d' AcquireRole événement.

Conditions préalables

L'activation du gestionnaire de rôles nécessite les `iam:CreateServiceLinkedRole` autorisations `iam:PutAccountProperties` et. La désactivation du gestionnaire de rôles nécessite une `iam:PutAccountProperties` autorisation. L'affichage du paramètre actuel nécessite une `iam:GetAccountProperties` autorisation. La politique AWS gérée `IAMFullAccess` inclut toutes ces autorisations. Pour connaître toutes les autorisations requises par le gestionnaire de rôles, consultezGérer l'accès au gestionnaire de rôles.

Comment activer et désactiver le gestionnaire de rôles

Vous pouvez activer ou désactiver le gestionnaire de rôles à l'aide de l' AWS API Console de gestion AWS AWS CLI, ou. Le gestionnaire de rôles est contrôlé par la propriété au niveau du compte`RoleManager/Enabled`, que vous définissez à l'aide de l'`PutAccountProperties`API. La valeur `true` active le gestionnaire de rôles ; le `false` désactive.

> **Note**
>
> Pour la plupart AWS des comptes, le gestionnaire de rôles est désactivé par défaut. Mais si vous avez créé votre compte à l'aide de notre nouvelle AWS expérience, le gestionnaire de rôles est activé par défaut et ne peut pas être désactivé tant que vous n'avez pas activé les fonctionnalités avancées. Pour plus d'informations, voir Activer les AWS fonctionnalités avancées.

Pour activer ou désactiver le gestionnaire de rôles (console)

Le gestionnaire de rôles peut être activé ou désactivé sur la page des paramètres du compte de la console IAM. Cette page indique également si le gestionnaire de rôles est activé pour le compte.

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la.
2. Dans le panneau de navigation, choisissez Paramètres du compte.
3. Dans la section du gestionnaire de rôles, choisissez Activer ou Désactiver.

Les rôles créés par le gestionnaire de rôles ne sont pas supprimés lorsque vous désactivez le gestionnaire de rôles.

Pour activer ou désactiver le gestionnaire de rôles (AWS CLI)

Définissez la `RoleManager/Enabled` propriété sur `true` pour activer le gestionnaire de rôles ou `false` pour le désactiver.

- Pour activer le gestionnaire de rôles, exécutez `put-account-properties` :

```
aws iam put-account-properties --properties RoleManager/Enabled=true
```

- Pour désactiver le gestionnaire de rôles, exécutez `put-account-properties` :

```
aws iam put-account-properties --properties RoleManager/Enabled=false
```

- Pour savoir si le gestionnaire de rôles est activé pour le compte, exécutez `get-account-properties` :

```
aws iam get-account-properties
```

Pour activer ou désactiver le gestionnaire de rôles (AWS API)

- Pour activer ou désactiver le gestionnaire de rôles, appelez PutAccountPropertiesavec la `RoleManager/Enabled` propriété définie sur `true` ou`false`.
- Pour voir le réglage actuel, appelez GetAccountProperties.

Pour plus d'informations, voir PutAccountPropertieset GetAccountPropertiesdans le manuel de référence de l'API IAM.

AWS consoles de service prenant en charge le gestionnaire de rôles

Le gestionnaire de rôles prend en charge la création de rôles pour les consoles AWS de service suivantes :

- AWS Backup
- AWS CloudFormation
- Amazon CloudWatch
- Amazon EKS
- AWS Elastic Beanstalk
- Amazon EventBridge
- AWS Glue
- AWS Lambda
- Amazon RDS
- Amazon SageMaker Unified Studio
- AWS Secrets Manager
- AWS Step Functions

AWS continuera à ajouter un service d'assistance au fil du temps. Pour en savoir plus sur le modèle de rôle utilisé par chaque service, consultezVue d'ensemble des modèles de rôles.

Informations connexes

- Gérer l'accès au gestionnaire de rôles
- Appliquer les autorisations de moindre privilège à un rôle créé automatiquement
- Vue d'ensemble des modèles de rôles

Gérer l'accès au gestionnaire de rôles

Vous contrôlez l'accès au gestionnaire de rôles à deux niveaux. Les autorisations IAM déterminent qui, dans un compte, peut activer le gestionnaire de rôles et qui peut utiliser les rôles qu'il fournit. Dans une organisation, les politiques de contrôle des services (SCP) déterminent ce que les comptes membres peuvent faire.

Autorisations pour le gestionnaire de rôles

Pour activer ou désactiver le gestionnaire de rôles, vous devez disposer de cette `iam:PutAccountProperties` autorisation. La politique AWS gérée l'`IAMFullAccess`inclut.

Pour utiliser le gestionnaire de rôles, un utilisateur doit disposer des autorisations IAM pour les actions que le gestionnaire de rôles effectue en son nom. Le gestionnaire de rôles fournit des rôles via l'`AcquireRole`API, qui évalue chaque action IAM sous-jacente par rapport aux autorisations de l'utilisateur :

- Si un rôle correspondant au modèle existe déjà dans le compte, `AcquireRole` renvoie ce rôle. Cela nécessite l'autorisation `iam:GetRole`.
- S'il n'existe aucun rôle correspondant, `AcquireRole` créez-en un à partir du modèle. Cela nécessite l'`iam:CreateRole`autorisation, ainsi que les autorisations pour ce que le modèle définit : `iam:PutRolePolicy` si le modèle ajoute des politiques en ligne et `iam:AttachRolePolicy` si le modèle joint des politiques gérées.

Un utilisateur disposant d'un accès IAM complet dispose de toutes les autorisations requises par le gestionnaire de rôles. Pour voir les actions exactes requises par un modèle spécifique, récupérez le modèle `GetRoleTemplateVersion` et passez en revue ses politiques.

Lorsque le gestionnaire de rôles crée un rôle, il AWS CloudTrail enregistre la création comme un `AcquireRole` événement unique. L'événement indique qui a appelé l'opération, le modèle de rôle et les valeurs de paramètres qui ont été utilisés, ainsi que le rôle qui a été créé.

L'exemple suivant montre que le gestionnaire de rôles a appelé l'`AcquireRole`opération pour le compte d'un utilisateur qui a assumé le rôle`Admin`, en utilisant le modèle de `PowerUserRoleTemplate` rôle pour créer le rôle`PowerUserRole`.


```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA123EXAMPLE:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "arn": "arn:aws:sts::123456789012:assumed-role/Admin/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "accountId": "123456789012",
        "accessKeyId": "ASIAWZYMLEXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA123EXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/Admin",
                "accountId": "123456789012",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2026-08-11T18:18:50Z",
                "mfaAuthenticated": "false"
            }
        },
        "onBehalfOf": {
            "userId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "identityStoreArn": "arn:aws:identitystore::111122223333:identitystore/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
        }
    },
    "eventTime": "2026-08-11T18:23:00Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "AcquireRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/151.0.0.0 Safari/537.36",
```

```
    "requestParameters": {
        "templateArn": "arn:aws:iam::aws:role-template/iam.amazonaws.com/
PowerUserRoleTemplate:1",
        "templateMinorVersion": 0,
        "replacementValues": {
            "AWSServiceName": {
                "values": [
                    "iam.amazonaws.com"
                ]
            }
        }
    },
    "responseElements": {
        "role": {
            "path": "/",
            "roleName": "PowerUserRole",
            "roleId": "AROA987EXAMPLE",
            "arn": "arn:aws:iam::123456789012:role/PowerUserRole",
            "createDate": "2026-08-11T18:23:00Z",
            "assumeRolePolicyDocument": "%7B%22Version%22:%222012-10-17%22,
%22Statement%22:%5B%7B%22Effect%22:%22Allow%22,%22Principal%22:%7B%22Service%22:
%22iam.amazonaws.com%22%7D,%22Action%22:%22sts:AssumeRole%22%7D%5D%7D"
        }
    },
    "requestID": "fde3dac0-73ac-491c-876b-EXAMPLE89c57",
    "eventID": "c93332cf-1429-4aee-bcaa-EXAMPLE3790d",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "iam.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```


Gestionnaire de rôles de contrôle dans une organisation

Vous pouvez contrôler le gestionnaire de rôles au sein d'une organisation à l'aide d'une politique de contrôle AWS des services (SCP) de l'organisation. Les exemples suivants montrent comment

bloquer l'activation du gestionnaire de rôles, bloquer la création de rôles ou bloquer uniquement la création de rôles basée sur un modèle.

Ces exemples utilisent la stratégie de liste de refus, ce qui signifie que vous avez également besoin `FullAWSAccess` d'une autre politique autorisant les actions associées aux entités de votre organisation. Vous devez toujours accorder des autorisations à vos mandants à l'aide de politiques basées sur l'identité.

Pour empêcher les comptes membres d'activer le gestionnaire de rôles via la console ou l'API, refusez l'`iam:PutAccountProperties`autorisation. Nous vous recommandons de suivre cette politique lorsque vous créez de nouveaux comptes directement par le biais d'Organisations.

```
{
    "Sid": "DenyRoleManagerEnablement",
    "Effect": "Deny",
    "Action": "iam:PutAccountProperties",
    "Resource": "*",
    "Condition": {
        "ForAnyValue:StringEquals": {
            "iam:AccountPropertyNamespaces": "RoleManager"
        }
    }
}
```

Bloquer l'accès pour utiliser le gestionnaire de rôles

Pour bloquer l'utilisation du gestionnaire de rôles, utilisez la clé `iam:RoleTemplateARN` contextuelle, présente sur les actions IAM qui créent un rôle à partir d'un modèle. La politique suivante refuse toutes les actions IAM dont la clé de `iam:RoleTemplateARN` contexte comporte une valeur non nulle. Cette politique n'empêche pas la création de tous les rôles : elle bloque uniquement la création de rôles via le gestionnaire de rôles et la création de rôles basée sur des modèles via le SDK.

```
{
    "Sid": "DenyTemplatedRoleCreation",
    "Effect": "Deny",
    "Action": "iam:*",
    "Resource": "*",
    "Condition": {
        "Null": {
            "iam:RoleTemplateARN": false
```

```
        }
    }
}
```

Bloquer la création de rôles sauf via le gestionnaire de rôles

Pour autoriser la création de rôles uniquement via le gestionnaire de rôles, utilisez la politique suivante pour bloquer la création de rôles autres que ceux de gestionnaire de rôles. Cette politique permet toujours la création de rôles sur la base de modèles directement via le SDK.

```
{
    "Sid": "DenyNonTemplatedRoleCreation",
    "Effect": "Deny",
    "Action": [
        "iam:CreateRole",
        "iam:AttachRolePolicy",
        "iam:PutRolePolicy",
        "iam:PutRolePermissionsBoundary",
        "iam:TagRole"
    ],
    "Resource": "*",
    "Condition": {
        "Null": {
            "iam:RoleTemplateARN": true
        }
    }
}
```

Informations connexes

- Créez des rôles automatiquement avec le gestionnaire de rôles
- Appliquer les autorisations de moindre privilège à un rôle créé automatiquement
- Vue d'ensemble des modèles de rôles

Appliquer les autorisations de moindre privilège à un rôle créé automatiquement

Lors de la création d'une ressource dans une console de service, le gestionnaire de rôles fournit le rôle IAM dont la ressource a besoin. Pour certaines tâches, telles que l'exécution de votre propre code, vous AWS ne pouvez pas déterminer à l'avance quelles autorisations sont requises. Par conséquent, un rôle peut accorder plus d'autorisations que la ressource n'en a besoin. Une fois que

vous connaissez les autorisations dont la ressource a besoin, vous pouvez réduire les autorisations du rôle en conséquence.

La désactivation du gestionnaire de rôles n'affecte pas les rôles précédemment créés par le gestionnaire de rôles. Vous pouvez désactiver le gestionnaire de rôles sans réduire les autorisations, et vous pouvez réduire les autorisations d'un rôle sans désactiver le gestionnaire de rôles.

Quand appliquer les autorisations de moindre privilège

L'attribution à un rôle uniquement des autorisations dont sa ressource a besoin est une bonne pratique de sécurité conforme au principe du moindre privilège. Pour la plupart des rôles, le gestionnaire de rôles crée des rôles dotés d'autorisations bien définies. Pour certains rôles, tels que ceux relatifs aux ressources de calcul ou à la gestion de l'infrastructure cloud, il crée des rôles dotés de larges autorisations.

Lorsque vous êtes prêt à définir la portée des rôles créés par le gestionnaire de rôles, nous vous recommandons de commencer par les rôles associés à vos charges de travail et à vos ressources les plus sensibles, tels que tout ce qui concerne les informations personnelles ou les données de production. Pour en savoir plus, veuillez consulter la section Préparation des autorisations de moindre privilège.

Comment analyser les accès non utilisés

Avant de désactiver le gestionnaire de rôles, vous AWS recommande de IAM Access Analyzer créer un analyseur d'accès non utilisé dans votre compte. L'analyseur compare les actions qu'un rôle est autorisé à effectuer avec les actions qu'il a effectuées et signale les autorisations non utilisées. L'analyseur passe en revue les rôles IAM de votre compte, y compris les rôles que vous avez créés vous-même.

L'endroit où les résultats apparaissent dépend de la manière dont l'analyseur a été fourni :

- Analyseur fourni par le gestionnaire de rôles : lorsque vous AWS fournissez l'analyseur d'accès non utilisé pour votre compte (voir la note dans cette section), les résultats apparaissent sur la page Rôles de la console IAM. Pour chaque rôle, vous pouvez voir le nombre d'autorisations non utilisées et ouvrir une recommandation. Suivez la procédure suivante pour réduire les autorisations d'un rôle depuis la page Rôles. Une fois que vous avez désactivé le gestionnaire de rôles, la page des paramètres du compte comporte un bouton Afficher les rôles qui ouvre la page Rôles, où les résultats de chaque rôle apparaissent.
- Analyseur créé par vous-même : lorsque vous créez votre propre analyseur d'accès non utilisé, les résultats et les recommandations apparaissent dans la console IAM Access Analyzer. Cette

expérience est inchangée par le gestionnaire de rôles. Pour examiner les résultats d'accès non utilisés et y donner suite, consultez les résultats d'IAM Access Analyzer.

> **Note**
>
> Si vous avez AWS activé le gestionnaire de rôles pour votre compte créé à l'aide de la nouvelle AWS expérience, vous n'avez pas besoin de créer un analyseur la première fois que vous désactivez le gestionnaire de rôles. Une fois que vous avez activé les fonctionnalités avancées et désactivé le gestionnaire de rôles, AWS fournit un analyseur d'accès non utilisé sans frais supplémentaires pendant 90 jours. L'analyseur vous donne une visibilité sur les autorisations de rôle non utilisées et vous recommande de réduire la portée des politiques afin que vous puissiez mettre à jour les rôles de votre compte, y compris ceux créés par le gestionnaire de rôles, selon le principe du moindre privilège.

Conditions préalables

Pour consulter les résultats d'Access Analyzer, vous devez être autorisé à effectuer l'`access-analyzer:ListFindings`action. Pour appliquer une recommandation, vous devez être autorisé à exécuter `iam:CreatePolicyiam:AttachRolePolicy`, et `iam:DetachRolePolicy` sur le rôle.

Pour réduire les autorisations d'un rôle dans la console (analyseur fourni par le gestionnaire de rôles)

Ces étapes s'appliquent lorsque l'analyseur d'accès non utilisé pour votre compte est AWS fourni. Si vous avez créé votre propre analyseur, consultez et appliquez plutôt les recommandations dans la console IAM Access Analyzer. Consultez les résultats d'IAM Access Analyzer.

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la.
2. Dans le panneau de navigation, choisissez Rôles.
3. Sur la page Rôles, dans la colonne Résultats, cliquez sur le lien Autorisations non utilisées pour le rôle dont vous souhaitez réduire les autorisations.
4. Sur la page Réduire les autorisations de rôle, passez en revue les autorisations non utilisées signalées pour le rôle.
5. Comparez les autorisations recommandées avec les autorisations actuelles du rôle et examinez les actions que la recommandation supprime.
6. Choisissez Apporter des modifications.

Comment fonctionnent les autorisations recommandées

Sur la base des résultats de l'analyseur d'accès non utilisés, IAM crée une ou plusieurs politiques gérées par le client qui contiennent les autorisations recommandées. IAM associe les nouvelles politiques avant de détacher les anciennes, de sorte que le rôle conserve ses autorisations tout au long de la modification.

Par exemple, considérez un rôle d'exécution AWS Lambda. IAM Access Analyzer indique quelles autorisations du rôle ne sont pas utilisées et recommande une politique adaptée aux actions effectuées par le rôle, telles que l'écriture de journaux dans Amazon Logs. CloudWatch Vous examinez la recommandation en fonction de ce que fait la fonction, vous confirmez qu'elle couvre tous les besoins de la fonction et vous l'appliquez. Le rôle n'autorise alors que les actions dont la fonction a besoin.

> **Important**
>
> La recommandation est basée sur les 30 derniers jours d'activité. Une autorisation que la ressource utilise rarement sur un horizon temporel plus long, par exemple pour une tâche trimestrielle, peut apparaître comme non utilisée. Examinez la recommandation en fonction des besoins de votre charge de travail avant de l'appliquer.
>
> Une fois que vous avez modifié les politiques ou les politiques de confiance d'un rôle, le gestionnaire de rôles ne gère plus ce rôle. Le rôle ne détectera aucune modification de version du modèle de rôle initialement utilisé pour créer le rôle.

IAM limite le nombre de politiques gérées que vous pouvez associer à un rôle. Si l'application de la recommandation dépasse cette limite, la console vous en informe et n'applique pas la modification. Pour plus d'informations sur ce quota, consultezIAM et AWS STS quotas.

Informations connexes

- Créez des rôles automatiquement avec le gestionnaire de rôles
- Gérer l'accès au gestionnaire de rôles
- Vue d'ensemble des modèles de rôles

Vue d'ensemble des modèles de rôles

Un modèle de rôle est un modèle de rôle utilisé pour créer de nouveaux rôles. Le modèle définit la politique de confiance et les autorisations du rôle afin que vous obteniez un rôle prêt à l'emploi pour un service sans avoir à rédiger vous-même les politiques.

Le gestionnaire de rôles utilise des modèles de rôles AWS gérés, mais vous pouvez également utiliser des modèles de rôles sans gestionnaire de rôles. Cette page explique le fonctionnement des modèles de rôles et répertorie les modèles pour chaque service.

Fonctionnement des modèles de rôles

Un modèle de rôle contient une politique de confiance et des autorisations, ainsi qu'un Amazon Resource Name (ARN). La politique de confiance définit les principaux responsables qui peuvent assumer ce rôle. Les autorisations définissent ce que le rôle peut faire. Les versions des modèles sont immuables. Lors de la AWS mise à jour d'un modèle, il publie une nouvelle version plutôt que de modifier une version existante.

Les modèles de rôle peuvent accepter des paramètres supplémentaires pour personnaliser et définir les propriétés et les autorisations du rôle. Par exemple, un modèle peut prendre un `RoleName` pour le rôle qu'il crée ou un identifiant de ressource tel que `bucketName` a. Lorsque le gestionnaire de rôles crée un rôle dans la console, il applique les paramètres aux autorisations et à la politique de confiance associées au rôle. Lorsque vous créez un rôle à partir d'un modèle directement avec l'API IAM, vous fournissez ces paramètres.

Pour créer un rôle à partir d'un modèle, utilisez l'`iam:AcquireRole`API avec l'ARN du modèle et transmettez les paramètres requis par le modèle.

Les définitions d'un modèle de rôle peuvent changer d'une version à l'autre. Pour voir la politique de confiance et les autorisations actuelles pour un modèle, récupérez-le à `iam:GetRoleTemplateVersion` l'aide de l'ARN du modèle. AWS recommande d'inspecter le modèle avant de s'y fier, en particulier pour un modèle qui accorde un accès étendu.

Comment le gestionnaire de rôles utilise les modèles

Lors de la création d'une ressource dans une console de service prise en charge, le gestionnaire de rôles fournit un rôle à partir du modèle qui correspond au service et au cas d'utilisation. Le gestionnaire de rôles choisit le modèle et fournit les paramètres pour vous. Le résultat est le même rôle que celui que l'on vous appelleraiti`am:AcquireRole`. Pour activer le gestionnaire de rôles, consultezGérer l'accès au gestionnaire de rôles.

Répertoire des modèles de rôles

Le tableau suivant répertorie le modèle de rôle pour chaque service pris en charge par le gestionnaire de rôles, avec l'ARN du modèle et les paramètres qu'il prend en charge. Utilisez l'ARN pour créer un rôle à partir du modèle ou pour vérifier le rôle accordé par le responsable pour ce service. Les services sont classés par ordre alphabétique.

Service	Modèle de rôle	ARN du modèle	Parameters
AWS Backup	AWSBackupDefaultServiceRoleTemplate	`arn:aws:iam::aws:role-template/backup.amazonaws.com/AWSBackupDefaultServiceRoleTemplate:1`	AccountID, RoleName
AWS CloudFormation	PowerUserRoleTemplate	`arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1`	AWSServiceName, RoleName
Amazon EKS	AmazonEKSAutoClusterRoleTemplate	`arn:aws:iam::aws:role-template/eks.amazonaws.com/AmazonEKSAutoClusterRoleTemplate:1`	RoleName
Amazon EKS	AmazonEKSAutoNodeRoleTemplate	`arn:aws:iam::aws:role-template/ec2.amazonaws.com/AmazonEKSAutoNodeRoleTemplate:1`	RoleName
AWS Elastic Beanstalk	PowerUserRoleTemplate	`arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1`	AWSServiceName, RoleName
Amazon EventBridge	PowerUserRoleTemplate	`arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1`	AWSServiceName, RoleName
AWS Glue	AWSGlueCrawlerS3PathsRoleTemplate	`arn:aws:iam::aws:role-template/glue.amazonaws.com/`	RoleName, AccountId, s3path

Service	Modèle de rôle	ARN du modèle	Parameters
		AWSGlueCrawlerS3PathsRoleTemplate:1	
AWS Glue	AWSGluePowerUserRoleTemplate	arn:aws:iam::aws:role-template/glue.amazonaws.com/AWSGluePowerUserRoleTemplate:1	AccountID, RoleName
AWS Lambda	PowerUserRoleTemplate	arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1	AWSServiceName, RoleName
Amazon RDS	AmazonRDSDirectoryServiceIntegrationRoleTemplate	arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSDirectoryServiceIntegrationRoleTemplate:1	RoleName, AccountID
Amazon RDS	AmazonRDSExportToS3RoleTemplate	arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSExportToS3RoleTemplate:1	RoleName, AccountID, BucketName
Amazon RDS	AmazonRDSImportFromS3RoleTemplate	arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSImportFromS3RoleTemplate:1	CUSTOM_KMS_KEY, RoleName AccountID, BucketName, clé, KeyRegion, kms, préfixe AccountId KeyId
Amazon RDS	AmazonRDSMonitoringRoleTemplate	arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSMonitoringRoleTemplate:1	RoleName, AccountID

Service	Modèle de rôle	ARN du modèle	Parameters
Amazon RDS	AmazonRDSProxyRoleTemplate	`arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSProxyRoleTemplate:2`	PROXY_WITH_CUSTOM_KEY, PROXY_WITH_DB_CONNECTION, PROXY_WITH_SECRET, AccountID, db, DBUser, KeyID, région, SecreTid RoleName ResourceId
Amazon RDS	AmazonRDSSecretAccessForDMSRoleTemplate	`arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSSecretAccessForDMSRoleTemplate:1`	CUSTOM_KMS_KEY, AccountID, RoleName km, région, SecreTid KeyId
Amazon SageMaker Unified Studio	AmazonSageMakerAdminIAMPermissiveExecutionRoleTemplate	`arn:aws:iam::aws:role-template/datazone.amazonaws.com/AmazonSageMakerAdminIAMPermissiveExecutionRoleTemplate:1`	CMK_ENABLED, RoleName AccountID, clé, KeyRegion, km AccountId KeyId
Amazon SageMaker Unified Studio	AmazonSageMakerUserIAMPermissiveExecutionRoleTemplate	`arn:aws:iam::aws:role-template/datazone.amazonaws.com/AmazonSageMakerUserIAMPermissiveExecutionRoleTemplate:1`	RoleName, AccountID

Service	Modèle de rôle	ARN du modèle	Parameters
AWS Secrets Manager	AWSSecretsManagerRotationRoleTemplate	`arn:aws:iam::aws:role-template/secretsmanager.amazonaws.com/AWSSecretsManagerRotationRoleTemplate:1`	ADMIN_RESOURCE_ENABLED, CMK_ENABLED, RoleNameAccountID, AdminType, km, région, ResourceTypeKeyArn
Amazon CloudWatch	PowerUserRoleTemplate	`arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1`	AWSServiceName, RoleName

Informations connexes

- Créez des rôles automatiquement avec le gestionnaire de rôles
- Gérer l'accès au gestionnaire de rôles
- Référence de l'API IAM

Gestion des rôles IAM

Avant qu'un utilisateur, une application ou un service soit en mesure d'utiliser un rôle que vous avez créé, vous devez lui accorder les autorisations nécessaires pour changer de rôle. Pour accorder ces autorisations, vous pouvez utiliser n'importe quelle politique attachée à l'un des groupes ou des utilisateurs. Cette section décrit l'octroi d'autorisations aux utilisateurs pour utiliser un rôle. Il explique également comment l'utilisateur peut passer d'un rôle à un autre dans les Outils pour Windows PowerShell, le AWS Command Line Interface (AWS CLI) et l'`AssumeRole`API. Console de gestion AWS

> **Important**
>
> Si vous créez un rôle par programmation plutôt que dans la console IAM, vous avez la possibilité d'ajouter un chemin `Path` de 512 caractères au maximum à l'élément `RoleName` qui lui, est de 64 caractères au maximum. Toutefois, si vous avez l'intention d'utiliser un rôle avec la fonctionnalité Switch Role dans le Console de gestion AWS, la combinaison `Path` `RoleName` ne peut pas dépasser 64 caractères.

Rubriques

- Afficher l'accès au rôle
- Générer une politique sur la base d'informations d'accès
- Octroi d'autorisations à un utilisateur pour endosser un rôle
- Accorder à un utilisateur l'autorisation de transmettre un rôle à un AWS web
- Révocation des informations d'identification de sécurité temporaires d'un rôle IAM
- Mise à jour d'un rôle lié à un service
- Mise à jour d'une politique d'approbation de rôle
- Mettre à jour les autorisations pour un rôle
- Mise à jour des paramètres pour un rôle
- Suppression de rôles ou de profils d'instance

Afficher l'accès au rôle

Avant de modifier les autorisations d'un rôle, vous devez passer en revue ses activités récentes au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Générer une politique sur la base d'informations d'accès

Vous pouvez parfois octroyer plus d'autorisations à une entité IAM (utilisateur ou rôle) qu'elle n'en a besoin. Pour affiner les autorisations que vous octroyez, vous pouvez générer une politique IAM basée sur l'activité d'accès d'une entité. IAM Access Analyzer examine vos AWS CloudTrail journaux et génère un modèle de politique contenant les autorisations qui ont été utilisées par l'entité dans

la plage de dates que vous avez spécifiée. Vous pouvez utiliser le modèle pour créer une politique gérée avec des autorisations affinées, puis l'attacher à l'entité IAM. Ainsi, vous accordez uniquement les autorisations dont l'utilisateur ou le rôle a besoin pour interagir avec les AWS ressources correspondant à votre cas d'utilisation spécifique. Pour en savoir plus, consultez Génération d'une politique de l'analyseur d'accès IAM.

Octroi d'autorisations à un utilisateur pour endosser un rôle

Lorsqu'un administrateur crée un rôle pour un accès intercompte, il établit une relation d'approbation entre le compte propriétaire du rôle et des ressources (compte d'approbation) et le compte qui contient les utilisateurs (compte approuvé). Pour cela, l'administrateur du compte d'approbation spécifie le numéro du compte approuvé en tant que `Principal` dans la politique d'approbation du rôle. Cela permet potentiellement à n'importe quel utilisateur du compte approuvé d'endosser le rôle. Pour achever la configuration, l'administrateur du compte approuvé doit accorder l'autorisation d'endosser ce rôle à des groupes ou utilisateurs spécifiques du compte.

Pour accorder l'autorisation de passer à un rôle

1. En tant qu'administrateur du compte approuvé, créez une nouvelle politique pour l'utilisateur ou modifiez une politique existante pour y ajouter les éléments requis. Pour en savoir plus, consultez Création ou modification de la politique.
2. Choisissez ensuite la manière dont vous souhaitez partager les informations du rôle :

   - Lien du rôle : envoyez aux utilisateurs un lien qui les dirige vers la page Switch Role (Changer de rôle) dans laquelle tous les détails sont déjà remplis.
   - ID ou alias du compte : donnez à chaque utilisateur le nom du rôle ainsi que l'ID ou l'alias du compte. L'utilisateur accède ensuite à la page Switch Role (Changer de rôle) et ajoute les détails manuellement.

   Pour en savoir plus, consultez Fourniture d'informations à l'utilisateur.

Notez que vous ne pouvez changer de rôle que lorsque vous vous connectez en tant qu'utilisateur IAM, rôle ou SAML-federated rôle fédéré d'identité Web. Vous ne pouvez pas changer de rôle lorsque vous vous connectez en tant qu' Utilisateur racine d'un compte AWS.

Important

Vous ne pouvez pas passer d'un rôle Console de gestion AWS à un rôle nécessitant une ExternalId valeur. Vous ne pouvez basculer vers un tel rôle qu'en appelant l'API `AssumeRole` qui prend en charge le paramètre `ExternalId`.

Remarques

- Cette rubrique décrit les politiques applicables aux utilisateurs car, en fin de compte, il s'agit ici d'accorder à un utilisateur les autorisations nécessaires pour effectuer une tâche. Cependant, nous vous déconseillons d'octroyer des autorisations directement à un utilisateur individuel. Lorsqu'un utilisateur assume un rôle, il reçoit également les autorisations associées.
- Lorsque vous changez de rôle dans le Console de gestion AWS, la console utilise toujours vos informations d'identification d'origine pour autoriser le changement. Cela s'applique que vous vous connectiez en tant qu'utilisateur IAM, en tant que SAML-federated rôle ou en tant que rôle fédéré d'identité Web. Par exemple, si vous basculez sur RoleA, IAM utilise les informations d'identification du compte utilisateur ou du rôle fédéré d'origine pour déterminer si vous êtes autorisé à assumer le rôle RoleA. Si vous essayez ensuite de basculer sur RoleB pendant que vous utilisez RoleA, vos informations d'identification d'utilisateur ou du rôle fédéré d'origine sont utilisées pour autoriser votre tentative. Les informations d'identification de RoleA (Rôle A) ne sont pas utilisées pour cette action.

Rubriques

- Création ou modification de la politique
- Fourniture d'informations à l'utilisateur

Création ou modification de la politique

Une politique qui accorde à un utilisateur l'autorisation d'endosser un rôle doit inclure une instruction avec l'effet `Allow` sur les éléments suivants :

- L'action `sts:AssumeRole`
- L'Amazon Resource Name (ARN) du rôle dans un élément `Resource`

Les utilisateurs bénéficiant de la politique sont autorisés à endosser les rôles figurant sur la ressource (via l'appartenance à un groupe ou directement).

> **Note**
>
> Si `Resource` est définie sur `*`, l'utilisateur peut endosser n'importe quel rôle dans n'importe quel compte faisant confiance au compte de l'utilisateur. (En d'autres termes, la politique de confiance du rôle spécifie le compte de l'utilisateur en tant que `Principal`). La bonne pratique consiste à appliquer le principe du moindre privilège et à spécifier l'ARN complet uniquement pour les rôles dont l'utilisateur a besoin.

L'exemple suivant illustre une politique qui permet à l'utilisateur d'endosser des rôles dans un seul compte. En outre, la politique utilise un caractère générique (*) pour spécifier que l'utilisateur ne peut passer à un rôle que si le nom du rôle commence par les lettres `Test`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRole",
        "Resource": "arn:aws:iam::111122223333:role/Test*"
    }
}
```

> **Note**
>
> Les autorisations que le rôle octroie à l'utilisateur ne viennent pas s'ajouter aux autorisations dont il dispose déjà. Lorsqu'un utilisateur endosse un rôle, il abandonne temporairement ses autorisations d'origine, de manière à adopter celles accordées par le rôle. Lorsqu'il quitte le rôle, ses autorisations d'origine sont automatiquement restaurées. Par exemple, supposons que les autorisations de l'utilisateur lui permettent d'utiliser des instances Amazon EC2, mais que la politique des autorisations du rôle n'accorde pas ces autorisations. Dans ce cas, lorsque vous utilisez le rôle, l'utilisateur peut ne pas utiliser les instances Amazon EC2 dans

la console. En outre, les informations d'identification temporaires obtenues via `AssumeRole` ne fonctionnent pas par programmation avec les instances Amazon EC2.

Fourniture d'informations à l'utilisateur

Une fois que vous avez créé un rôle et accordé à l'utilisateur les autorisations requises pour l'endosser, vous devez également fournir à l'utilisateur les éléments suivants :

- Nom du rôle
- ID ou alias du compte qui contient le rôle

Pour faciliter l'accès aux utilisateurs, vous pouvez leur envoyer un lien préconfiguré contenant l'ID du compte et le nom du rôle. Vous pouvez voir le lien du rôle après avoir terminé l'assistant Créer un rôle en sélectionnant la bannière Afficher le rôle ou sur la page Résumé du rôle pour n'importe quel rôle inter-comptes.

Vous pouvez également utiliser le format suivant pour créer le lien manuellement. Remplacez les deux paramètres de l'exemple suivant par l'ID ou l'alias de votre compte et le nom du rôle.

`https://signin.aws.amazon.com/switchrole?account=`*`your_account_ID_or_alias`*`&roleName=`*`optional_path/role_name`*

Nous vous recommandons de diriger vos utilisateurs vers la rubrique Basculer d'un utilisateur à un rôle IAM (console) pour les guider à travers le processus. Pour résoudre les problèmes courants que vous pouvez rencontrer lorsque vous endossez un rôle, consultez Je ne parviens pas à endosser un rôle.

Considérations

- Si vous créez le rôle par programmation, vous pouvez définir un chemin et un nom. Dans ce cas, vous devez fournir le chemin complet et le nom du rôle à vos utilisateurs afin qu'ils les saisissent sur la page Switch Role (Changer de rôle) de la Console de gestion AWS. Par exemple : `division_abc/subdivision_efg/role_XYZ`.
- Si vous créez le rôle par programmation, vous pouvez ajouter un `Path` de 512 caractères au maximum , ainsi qu'un `RoleName`. Le nom du rôle peut comporter jusqu'à 64 caractères. Toutefois, pour utiliser un rôle avec la fonctionnalité Changer de rôle dans le Console de gestion AWS, la combinaison `Path RoleName` ne peut pas dépasser 64 caractères.

- Pour des raisons de sécurité, vous pouvez consulter AWS CloudTrail les journaux pour savoir qui a effectué une action AWS. Vous pouvez utiliser la clé de condition `sts:SourceIdentity` dans la politique d'approbation de rôle pour exiger des utilisateurs qu'ils spécifient une identité lorsqu'ils endossent un rôle. Par exemple, vous pouvez exiger que les utilisateurs IAM spécifient leur propre nom d'utilisateur comme identité de source. Cela peut vous aider à déterminer quel utilisateur a effectué une action spécifique dans AWS. Pour de plus amples informations, veuillez consulter `sts:SourceIdentity`. Vous pouvez utiliser `sts:RoleSessionName` pour exiger des utilisateurs qu'ils spécifient un nom de session lorsqu'ils endossent un rôle. Cela peut vous aider à différencier les sessions de rôle lorsqu'un rôle est utilisé par différents principaux.

Accorder à un utilisateur l'autorisation de transmettre un rôle à un AWS web

Pour configurer de nombreux AWS services, vous devez leur transmettre un rôle IAM. Cela autorise le service à assumer ultérieurement le rôle et à effectuer des actions en votre nom. Pour la plupart des services, il vous suffit de transférer le rôle au service une fois au cours de la configuration, et non chaque fois que le service assume le rôle. Par exemple, supposons que vous avez une application s'exécutant sur une instance Amazon EC2. Cette application a besoin d'informations d'identification temporaires pour l'authentification et d'autorisations pour autoriser l'application à exécuter des actions dans AWS. Lorsque vous configurez l'application, vous devez transmettre un rôle à Amazon EC2 à utiliser avec l'instance qui fournit ces informations d'identification. Vous définissez les autorisations pour les applications s'exécutant sur l'instance en attachant une politique IAM au rôle. L'application endosse le rôle chaque fois qu'elle doit effectuer les actions qui sont autorisées par le rôle.

Pour transmettre un rôle (et ses autorisations) à un AWS service, un utilisateur doit disposer des autorisations nécessaires pour transmettre le rôle au service. Cela permet aux administrateurs de s'assurer que seuls les utilisateurs autorisés peuvent configurer un service avec un rôle qui accorde des autorisations. Pour autoriser un utilisateur à transmettre un rôle à un AWS service, vous devez accorder l'`PassRole`autorisation à l'utilisateur, au rôle ou au groupe IAM de l'utilisateur.

> **Warning**
>
> - Vous ne pouvez utiliser cette `PassRole` autorisation que pour transmettre un rôle IAM à un service qui partage le même AWS compte. Pour transmettre un rôle du compte A à un service du compte B, vous devez d'abord créer un rôle IAM dans le compte B qui peut assumer le rôle du compte A, puis le rôle du compte B peut être transmis au service. Pour en savoir plus, consultez Accès intercompte aux ressources dans IAM.

- N'essayez pas de contrôler les personnes susceptibles de faire passer un rôle en étiquetant ce rôle, puis en utilisant la clé de condition `ResourceTag` dans une politique avec l'action `iam:PassRole`. Cette approche ne produit pas des résultats fiables.

Lors de la définition de l'autorisation `PassRole`, vous devez vous assurer qu'un utilisateur ne transfère pas un rôle ayant plus d'autorisations que ce que vous souhaitez pour l'utilisateur. Par exemple, Alice n'est peut-être pas autorisée à effectuer des actions Amazon S3. Si Alice pouvait transmettre un rôle à un service autorisant les actions Amazon S3, le service pourrait effectuer des actions Amazon S3 pour le compte d'Alice lors de l'exécution de la tâche.

Lorsque vous spécifiez un rôle lié à un service, vous devez également posséder l'autorisation de transmettre ce rôle au service. Certains services créent automatiquement un rôle lié à un service dans votre compte lorsque vous effectuez une action dans ce service. Par exemple, Amazon EC2 Auto Scaling crée automatiquement le rôle lié au service `AWSServiceRoleForAutoScaling` la première fois que vous créez un groupe Auto Scaling. Si vous essayez de spécifier le rôle lié au service lorsque vous créez un groupe Auto Scaling sans l'autorisation `iam:PassRole`, vous recevez une erreur. Si vous ne spécifiez pas explicitement le rôle, l'autorisation `iam:PassRole` n'est pas requise et la valeur par défaut est d'utiliser le rôle `AWSServiceRoleForAutoScaling` pour toutes les opérations effectuées sur ce groupe. Pour savoir quels services prennent en charge les rôles liés à un service, veuillez consulter AWS services qui fonctionnent avec IAM. Pour savoir quels services créent automatiquement un rôle lié à un service lorsque vous effectuez une action dans ce service, choisissez le lien Oui et consultez la documentation relative au rôle lié à un service pour le service.

Un utilisateur peut transmettre un ARN de rôle comme paramètre dans n'importe quelle opération d'API qui utilise le rôle pour attribuer des autorisations au service. Le service vérifie ensuite si cet utilisateur dispose de l'autorisation `iam:PassRole`. Pour que l'utilisateur transfère uniquement les rôles approuvés, vous pouvez filtrer l'autorisation `iam:PassRole` à l'aide de l'élément `Resources` de l'instruction de politique IAM.

Vous pouvez utiliser `Condition` cet élément dans une politique JSON pour tester la valeur des clés incluses dans le contexte de toutes les AWS demandes. Pour en savoir plus sur l'utilisation des clés de condition dans une politique, consultez la section Éléments de la politique JSON IAM : Condition. La clé de condition `iam:PassedToService` peut être utilisée pour spécifier le principal de service du service auquel un rôle peut être transmis. Pour en savoir plus sur l'utilisation de la clé de `iam:PassedToService` condition dans une politique, consultez iam : PassedToService.

Exemple 1

Supposons que vous vouliez accorder à un utilisateur la possibilité de transférer l'un des ensembles de rôles approuvés au service Amazon EC2 lors du lancement d'une instance. Vous avez besoin de trois éléments :

- Une politique d'autorisations IAM attachée au rôle qui détermine les opérations pouvant être exécutées par le rôle. Limitez les autorisations aux seules actions nécessaires pour le rôle et aux seules ressources dont le rôle a besoin pour exécuter ces actions. Vous pouvez utiliser une politique d' AWS autorisations IAM gérée ou créée par le client.

  JSON

  ```
  {
      "Version":"2012-10-17",
      "Statement": {
          "Effect": "Allow",
          "Action": [ "A list of the permissions the role is allowed to use" ],
          "Resource": [ "A list of the resources the role is allowed to access" ]
      }
  }
  ```

- Une politique d'approbation pour le rôle, qui permet au service d'endosser le rôle. Par exemple, vous pourriez attacher la politique d'approbation suivante au rôle avec l'action `UpdateAssumeRolePolicy`. Cette politique d'approbation permet à Amazon EC2 d'utiliser le rôle et les autorisations attachées au rôle.

  JSON

  ```
  {
      "Version":"2012-10-17",
      "Statement": {
          "Sid":
   "TrustPolicyStatementThatAllowsEC2ServiceToAssumeTheAttachedRole",
          "Effect": "Allow",
          "Principal": { "Service": "ec2.amazonaws.com" },
         "Action": "sts:AssumeRole"
      }
  }
  ```

- Une politique d'autorisations IAM attachée à l'utilisateur IAM, qui permet à l'utilisateur de transférer uniquement les rôles qui sont approuvés. Vous ajoutez généralement `iam:GetRole` à `iam:PassRole` pour que l'utilisateur puisse obtenir les détails du rôle à transférer. Dans cet

exemple, l'utilisateur peut transférer uniquement les rôles qui existent dans le compte spécifié avec des noms commençant par l'interface `EC2-roles-for-XYZ-` :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::111122223333:role/EC2-roles-for-XYZ-*"
        }
    ]
}
```

L'utilisateur peut maintenant démarrer une instance Amazon EC2 avec un rôle affecté. Les applications qui s'exécutent sur l'instance peuvent accéder à des informations d'identification temporaires pour le rôle via les métadonnées du profil d'instance. Les politiques d'autorisations attachées au rôle déterminent ce que peut faire l'instance.

Exemple 2

Amazon Relational Database Service (Amazon RDS) prend en charge une fonction appelée surveillance améliorée. Cette fonctionnalité permet à Amazon RDS de surveiller une instance de base de données grâce à un agent. Il permet également à Amazon RDS d'enregistrer des métriques dans Amazon CloudWatch Logs. Pour activer cette fonctionnalité, vous devez créer un rôle de service pour accorder à Amazon RDS des autorisations pour surveiller et écrire des métriques dans vos journaux.

Pour créer un rôle pour la surveillance améliorée Amazon RDS

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Cliquez sur Rôles, puis sur Créer un rôle.

3. Choisissez le type AWS de rôle de service, puis, dans Cas d'utilisation pour les autres Services AWS, choisissez le service RDS. Choisissez RDS - Enhanced Monitoring (RDS - Surveillance améliorée), puis Next (Suivant).
4. Choisissez la politique AmazonRDSEnhancedMonitoringRole d'autorisations.
5. Choisissez Suivant.
6. Pour Role name (Nom du rôle), saisissez un nom de rôle vous permettant d'identifier le but de ce rôle. Les noms de rôles doivent être uniques au sein de votre Compte AWS. Lorsqu'un nom de rôle est utilisé dans une politique ou dans le cadre d'un ARN, il est sensible à la casse. Lorsque le nom d'un rôle apparaît aux clients dans la console, par exemple lors du processus de connexion, il n'est pas sensible à la casse. Différentes entités peuvent référencer le rôle et il n'est donc pas possible de modifier son nom après sa création.
7. (Facultatif) Pour Description, saisissez une description pour le nouveau rôle.
8. (Facultatif) Ajoutez des métadonnées à l'utilisateur en associant les balises sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Tags pour Gestion des identités et des accès AWS resources.
9. Passez en revue les informations du rôle, puis choisissez Créer un rôle.

Le rôle obtient automatiquement une politique d'approbation qui accorde au service `monitoring.rds.amazonaws.com` les autorisations nécessaires pour endosser le rôle. Une fois cette opération effectuée, Amazon RDS peut exécuter toutes les actions que la politique `AmazonRDSEnhancedMonitoringRole` autorise.

L'utilisateur auquel vous souhaitez accéder à Enhanced Monitoring a besoin d'une politique qui inclut une instruction lui permettant de répertorier les rôles RDS et une instruction lui permettant de transmettre le rôle, comme suit. Utilisez votre numéro de compte et remplacez le nom du rôle par le nom fourni à l'étape 6.

```
    {
      "Sid": "PolicyStatementToAllowUserToListRoles",
      "Effect": "Allow",
      "Action": ["iam:ListRoles"],
      "Resource": "*"
    },
    {
        "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
        "Effect": "Allow",
        "Action": [ "iam:PassRole" ],
```

```
        "Resource": "arn:aws:iam::account-id:role/RDS-Monitoring-Role"
    }
```

Vous pouvez combiner cette instruction avec des instructions d'une autre politique ou la placer dans sa propre politique. Au lieu de spécifier que l'utilisateur peut transférer un rôle commençant par `RDS-`, vous pouvez remplacer le nom du rôle dans l'ARN d'une ressource par un caractère générique, comme suit.

```
        "Resource": "arn:aws:iam::account-id:role/RDS-*"
```

`iam : PassRole` actions dans AWS CloudTrail journaux

`PassRole`n'est pas un appel d'API. `PassRole`est une autorisation, ce qui signifie qu'aucun CloudTrail journal n'est généré pour IAM`PassRole`. Pour vérifier quels rôles sont transmis Services AWS à quels rôles CloudTrail, vous devez consulter le CloudTrail journal qui a créé ou modifié la AWS ressource recevant le rôle. Par exemple, un rôle est transmis à une AWS Lambda fonction lors de sa création. le journal répertoriant l'action `CreateFunction` affichera l'enregistrement du rôle qui a été transmis à la fonction.

Révocation des informations d'identification de sécurité temporaires d'un rôle IAM

> **Warning**
>
> Si vous suivez les étapes indiquées sur cette page, tous les utilisateurs dont les sessions ont été créées en assumant le rôle se voient refuser l'accès à toutes les AWS actions et ressources. Cela peut entraîner le risque pour les utilisateurs de perdre leur travail non sauvegardé.

Lorsque vous autorisez les utilisateurs à accéder Console de gestion AWS à une session de longue durée (12 heures, par exemple), leurs informations d'identification temporaires n'expirent pas aussi rapidement. Si les utilisateurs exposent par inadvertance leurs informations d'identification à un tiers non autorisé, celui-ci dispose d'un accès pendant la durée de la session. Cependant, vous pouvez révoquer immédiatement toutes les autorisations aux informations d'identification du rôle émises avant un moment donné, si nécessaire. Toutes les informations d'identification temporaires pour ce rôle qui ont été émises avant le moment spécifié deviennent non valides. Cela force tous les utilisateurs à s'authentifier de nouveau et demande de nouvelles informations d'identification.

> **Note**
>
> Vous ne pouvez pas révoquer la session d'un rôle lié à un service.

Lorsque vous révoquez des autorisations pour un rôle à l'aide de la procédure décrite dans cette rubrique, AWS vous associez une nouvelle politique intégrée au rôle qui refuse toutes les autorisations pour toutes les actions. Il inclut une condition qui applique les restrictions uniquement si l'utilisateur a endossé le rôle avant le moment où vous avez révoqué les autorisations. Si l'utilisateur endosse le rôle après la révocation des autorisations, la politique de rejet ne s'applique pas à cet utilisateur.

Pour plus d'informations sur le refus d'accès, veuillez consulter la rubrique Désactivation des autorisations affectées aux informations d'identification de sécurité temporaires.

> **Important**
>
> Cette politique de rejet s'applique à tous les utilisateurs du rôle spécifié, pas seulement à ceux dont les sessions de console durent plus longtemps.

Autorisations minimales pour révoquer les autorisations de session d'un rôle

Pour révoquer les autorisations de session d'un rôle avec succès, vous devez disposer de l'autorisation `PutRolePolicy` pour ce rôle. Cela vous permet d'attacher la politique en ligne `AWSRevokeOlderSessions` au rôle.

Révocation des autorisations de session

Vous pouvez révoquer les autorisations de session d'un rôle afin de refuser toutes les autorisations à tout utilisateur qui a endossé le rôle.

> **Note**
>
> Vous ne pouvez pas modifier les rôles qui ont été créés dans IAM à partir des jeux de permissions d'IAM IAM Identity Center. Vous devez révoquer la session d'ensemble d'autorisations active pour un utilisateur dans IAM Identity Center. Pour plus d'informations, consultez Révocation des sessions de rôle IAM actives créées par des ensembles d'autorisations dans le Guide de l'utilisateur d'IAM Identity Center.

Pour refuser immédiatement toutes autorisations à tout utilisateur actuel des informations d'identification du rôle

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, sélectionnez Rôles, puis le nom (pas la case à cocher) du rôle dont vous voulez révoquer les autorisations.
3. Sur la page Récapitulatif du rôle sélectionné, choisissez l'onglet Révoquer les sessions.
4. Dans l'onglet Révoquer les sessions, choisissez Révoquer les sessions actives.
5. AWS vous demande de confirmer l'action. Cochez la case Je reconnais que je révoque toutes les sessions actives pour ce rôle. et choisissez Révoquer les sessions actives dans la boîte de dialogue.

   IAM attache ensuite une politique nommée `AWSRevokeOlderSessions` au rôle. Après avoir sélectionné Révoquer les sessions actives, la politique refuse tout accès aux utilisateurs qui ont assumé le rôle dans le passé ainsi qu'environ 30 secondes dans le futur. Ce choix du temps futur tient compte du délai de propagation de la politique afin de traiter une nouvelle session acquise ou renouvelée avant que la politique mise à jour ne soit en vigueur dans une région donnée. Tout utilisateur qui endosse le rôle plus de 30 secondes environ après que vous avez choisi Révoquer les sessions actives n'est pas affecté. Pour savoir pourquoi les modifications ne sont pas toujours visibles immédiatement, consultez la section Les modifications que j'apporte ne sont pas toujours visibles immédiatement.

> **Note**
>
> Si vous choisissez Révoquer les sessions actives ultérieurement, l'horodatage de la politique sera actualisé et elle refusera à nouveau toutes les autorisations à tout utilisateur ayant endossé le rôle avant la nouvelle heure spécifiée.

Les utilisateurs valides dont les sessions sont révoquées de cette manière doivent obtenir des informations d'identification temporaires pour qu'une nouvelle session continue de fonctionner. Les informations d'identification sont mises en AWS CLI cache jusqu'à leur expiration. Pour forcer la CLI à supprimer et actualiser les informations d'identification mises en cache qui ne sont plus valides, exécutez l'une des commandes suivantes :

Linux, macOS ou Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

Révocation des autorisations de session avant une heure spécifiée

Vous pouvez également révoquer les autorisations de session à tout moment de votre choix à l'aide du SDK AWS CLI ou afin de spécifier une valeur pour la `aws : TokenIssueTime` clé dans l'élément Condition d'une politique.

Cette politique refuse toutes les autorisations lorsque la valeur de `aws:TokenIssueTime` est antérieure aux date et heure spécifiées. La valeur de `aws:TokenIssueTime` correspond à l'exacte à laquelle les informations d'identification de sécurité temporaires ont été créées. La `aws:TokenIssueTime` valeur n'est présente que dans le contexte des AWS demandes signées avec des informations d'identification de sécurité temporaires, de sorte que l'instruction Deny de la politique n'affecte pas les demandes signées avec les informations d'identification à long terme de l'utilisateur IAM.

Cette politique peut également être attachée à un rôle. Dans ce cas, la politique affecte uniquement les informations d'identification de sécurité temporaires créées par le rôle avant la date et l'heure spécifiées.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {"aws:TokenIssueTime": "2014-05-07T23:47:00Z"}
    }
  }
}
```

Les utilisateurs valides dont les sessions sont révoquées de cette manière doivent obtenir des informations d'identification temporaires pour qu'une nouvelle session continue de fonctionner. Les informations d'identification sont mises en AWS CLI cache jusqu'à leur expiration. Pour forcer la CLI à supprimer et actualiser les informations d'identification mises en cache qui ne sont plus valides, exécutez l'une des commandes suivantes :

Linux, macOS ou Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

Mise à jour d'un rôle lié à un service

La méthode que vous utilisez pour modifier un rôle lié à un service dépend dudit service. Certains services peuvent vous permettre de modifier les autorisations d'un rôle lié à un service depuis la console, l'API ou la CLI. Cependant, une fois un rôle lié à un service créé, vous ne pouvez pas changer le nom d'un rôle car plusieurs entités peuvent faire référence au rôle. Vous pouvez modifier la description de n'importe quel rôle depuis la console IAM, l'API ou la CLI.

Pour plus d'informations concernant la prise en charge par les services des rôles liés à un service, consultez AWS services qui fonctionnent avec IAM et recherchez les services affichant Oui dans la colonne Rôle lié à un service. Pour savoir si le service prend en charge la modification du rôle lié à un service, choisissez le lien Oui pour afficher la documentation du rôle lié à ce service.

Modification de la description d'un rôle lié à un service (console)

Vous pouvez utiliser la console IAM pour modifier la description d'un rôle lié à un service.

Pour modifier la description d'un rôle lié à un service (console)

1. Dans le panneau de navigation de la console IAM, choisissez Rôles.
2. Choisissez le nom du rôle à modifier.
3. A l'extrême droite de Description du rôle, choisissez Edit (Modifier).
4. Saisissez une nouvelle description dans la zone et choisissez Save (Enregistrer).

Modification de la description d'un rôle lié à un service (AWS CLI)

Vous pouvez utiliser les commandes IAM depuis le AWS CLI pour modifier la description d'un rôle lié à un service.

Pour changer la description d'un rôle lié à un service (AWS CLI)

1. (Facultatif) Pour afficher la description actuelle d'un rôle, exécutez les commandes suivantes :

   ```
   aws iam get-role --role-name ROLE-NAME
   ```

   Utilisez le nom du rôle, pas l'ARN, pour faire référence aux rôles avec les commandes CLI. Par exemple, si un rôle a l'ARN : `arn:aws:iam::123456789012:role/myrole`, vous faites référence au rôle en tant que **`myrole`**.
2. Pour mettre à jour la description d'un rôle lié à un service, exécutez la commande suivante :

   ```
   aws iam update-role --role-name ROLE-NAME --description OPTIONAL-DESCRIPTION
   ```

Modification d'une description de rôle liée à un service (API)AWS

Vous pouvez utiliser l' AWS API pour modifier la description d'un rôle lié à un service.

Pour modifier la description d'un rôle lié à un service (API)AWS

1. (Facultatif) Pour afficher la description actuelle d'un rôle, appelez l'opération suivante et spécifiez le nom du rôle :

   AWS API : GetRole
2. Pour mettre à jour la description d'un rôle, appelez l'opération suivante et spécifiez le nom (et la description facultative) du rôle :

   AWS API : UpdateRole

Mise à jour d'une politique d'approbation de rôle

Pour modifier la personne qui peut endosser un rôle, vous devez modifier la politique d'approbation du rôle. Vous ne pouvez pas modifier la politique d'approbation d'un rôle lié à un service.

Remarques

- Si un utilisateur est répertorié comme principal dans la politique d'approbation d'un rôle, mais ne peut pas endosser le rôle, vérifiez les limites d'autorisations de l'utilisateur. Si une limite d'autorisation est définie pour l'utilisateur, elle doit autoriser l'action `sts:AssumeRole`.
- Pour permettre aux utilisateurs d'assumer à nouveau le rôle actuel au cours d'une session de rôle, spécifiez l'ARN du rôle ou l' Compte AWS ARN en tant que principal dans la politique de confiance des rôles. Services AWS qui fournissent des ressources de calcul telles qu'Amazon EC2, Amazon ECS, Amazon EKS et Lambda fournissent des informations d'identification temporaires et mettent automatiquement à jour ces informations d'identification. Cela garantit que vous disposez toujours d'un ensemble d'informations d'identification valide. Pour ces services, il n'est pas nécessaire d'endosser à nouveau le rôle actuel pour obtenir des informations d'identification temporaires. Toutefois, si vous avez l'intention de transférer des balises de session ou une politique de session, vous devez endosser à nouveau le rôle actuel.

Mise à jour d'une politique d'approbation de rôle (console)

Pour modifier la politique de confiance d'un rôle dans Console de gestion AWS

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation de la console IAM, choisissez Rôles.
3. Dans la liste des rôles de votre compte, choisissez le nom du rôle que vous souhaitez modifier.
4. Sélectionnez l'onglet Trust relationships (Relations d'approbation), puis Edit trust policy (Modifier la politique d'approbation).
5. Modifiez la politique d'approbation au besoin. Pour ajouter des principaux supplémentaires capables d'endosser le rôle, spécifiez-les dans l'élément `Principal`. Par exemple, l'extrait de politique suivant montre comment en référencer deux Comptes AWS dans l'`Principal`élément :

```
"Principal": {
  "AWS": [
```

```
    "arn:aws:iam::111122223333:root",
    "arn:aws:iam::444455556666:root"
  ]
},
```

Si vous spécifiez un principal dans un autre compte, l'ajout d'un compte à la politique d'approbation d'un rôle n'est qu'une partie de la procédure d'établissement de la relation d'approbation entre comptes. Par défaut, aucun utilisateur des comptes approuvés ne peut endosser le rôle. L'administrateur du compte nouvellement approuvé doit accorder aux utilisateurs l'autorisation d'endosser le rôle. Pour ce faire, l'administrateur doit créer ou modifier une politique attachée à l'utilisateur pour lui permettre d'accéder à l'action `sts:AssumeRole`. Pour plus d'informations, consultez la procédure suivante ou Octroi d'autorisations à un utilisateur pour endosser un rôle.

L'extrait de politique suivant montre comment référencer deux AWS services dans l'`Principal`élément :

```
"Principal": {
  "Service": [
    "opsworks.amazonaws.com",
    "ec2.amazonaws.com"
  ]
},
```

6. Après avoir modifié votre politique d'approbation, choisissez Mettre à jour la politique pour enregistrer vos modifications.

   Pour plus d'informations sur la syntaxe et la structure de la politique, consultez les sections Politiques et autorisations dans Gestion des identités et des accès AWS et Référence de l'élément de politique JSON IAM.

Pour autoriser les utilisateurs d'un compte externe approuvé à utiliser le rôle (console)

Pour plus d'informations sur cette procédure, consultez Octroi d'autorisations à un utilisateur pour endosser un rôle.

1. Connectez-vous à l'interface externe de confiance Compte AWS.

2. Décidez si vous devez attacher les autorisations à un utilisateur ou à un groupe. Dans le panneau de navigation de la console IAM, sélectionnez Users (Utilisateurs) ou User groups (Groupes d'utilisateurs) selon le cas.
3. Choisissez le nom de l'utilisateur ou du groupe auquel vous souhaitez accorder l'accès, puis sélectionnez l'onglet Autorisations.
4. Effectuez l'une des actions suivantes :

   - Pour modifier une politique gérée par un client, choisissez le nom de la politique, choisissez Modifier la politique, puis l'onglet JSON. Vous ne pouvez pas modifier une politique AWS gérée. AWS les politiques gérées apparaissent avec l' AWS icône (). Pour plus d'informations sur la différence entre les stratégies gérées AWS et les stratégies gérées par le client, consultez Politiques gérées et politiques en ligne.
   - Pour modifier une politique en ligne, choisissez la flèche en regard du nom de la politique et choisissez Modifier la politique.
5. Dans l'éditeur de politiques, ajoutez un nouvel élément `Statement` spécifiant ce qui suit :

   ```
   {
     "Effect": "Allow",
     "Action": "sts:AssumeRole",
     "Resource": "arn:aws:iam::ACCOUNT-ID:role/ROLE-NAME"
   }
   ```

   Remplacez l'ARN dans l'instruction par celui du rôle que l'utilisateur peut endosser.
6. Suivez les invites à l'écran pour terminer de modifier la politique.

Mettre à jour une politique de confiance relative aux rôles (AWS CLI)

Vous pouvez utiliser le AWS CLI pour changer qui peut assumer un rôle.

Pour modifier la politique de confiance d'un rôle (AWS CLI)

1. (Facultatif) Si vous ne connaissez pas le nom du rôle que vous souhaitez modifier, exécutez la commande suivante pour répertorier les rôles de votre compte :

   - aws iam list-roles

2. (Facultatif) Pour afficher la politique d'approbation actuelle d'un rôle, exécutez la commande suivante :

   - aws iam get-role

3. Pour modifier les principaux approuvés ayant accès au rôle, créez un fichier texte avec la politique d'approbation mise à jour. Vous pouvez utiliser l'éditeur de texte pour créer la politique.

   Par exemple, la politique d'approbation suivante illustre comment référencer deux Comptes AWS dans l'élément `Principal`. Cela permet à des utilisateurs séparés Comptes AWS de jouer ce rôle.

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": {
           "Effect": "Allow",
           "Principal": {"AWS": [
               "arn:aws:iam::111122223333:root",
               "arn:aws:iam::444455556666:root"
           ]},
           "Action": "sts:AssumeRole"
       }
   }
   ```

   Si vous spécifiez un principal dans un autre compte, l'ajout d'un compte à la politique d'approbation d'un rôle n'est qu'une partie de la procédure d'établissement de la relation d'approbation entre comptes. Par défaut, aucun utilisateur des comptes approuvés ne peut endosser le rôle. L'administrateur du compte nouvellement approuvé doit accorder aux utilisateurs l'autorisation d'endosser le rôle. Pour ce faire, l'administrateur doit créer ou modifier une politique attachée à l'utilisateur pour lui permettre d'accéder à l'action `sts:AssumeRole`. Pour plus d'informations, consultez la procédure suivante ou Octroi d'autorisations à un utilisateur pour endosser un rôle.

4. Pour utiliser le fichier créé afin de mettre à jour la politique de confiance, exécutez la commande suivante :

   - aws iam update-assume-role-policy

Pour autoriser les utilisateurs d'un compte externe approuvé à utiliser le rôle (AWS CLI)

Pour plus d'informations sur cette procédure, consultez Octroi d'autorisations à un utilisateur pour endosser un rôle.

1. Créez un fichier JSON contenant une politique d'autorisations qui accorde des autorisations permettant d'endosser le rôle. Par exemple, la politique suivante contient les autorisations nécessaires minimales :

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": {
           "Effect": "Allow",
           "Action": "sts:AssumeRole",
           "Resource": "arn:aws:iam::111122223333:role/ROLE-NAME"
       }
   }
   ```

   Remplacez l'ARN dans l'instruction par celui du rôle que l'utilisateur peut endosser.
2. Exécutez la commande suivante pour charger le fichier JSON contenant la politique de confiance dans IAM :

   - aws iam create-policy

   Le résultat de cette commande inclut l'ARN de la politique. Notez cet ARN, car vous aurez besoin de l'utiliser ultérieurement.
3. Décidez à quel utilisateur ou groupe attacher la politique. Si vous ne connaissez pas le nom de l'utilisateur ou du groupe concerné, utilisez les commandes suivantes pour répertorier les utilisateurs ou les groupes de votre compte :

   - aws iam list-users
   - aws iam list-groups
4. Utilisez l'une des commandes suivantes pour attacher la politique créée dans l'étape précédente à l'utilisateur ou au groupe :

- aws iam attach-user-policy
- aws iam attach-group-policy

Mettre à jour une politique de confiance relative aux rôles (AWS API)

Vous pouvez utiliser l' AWS API pour modifier les personnes autorisées à assumer un rôle.

Pour modifier la politique de confiance d'un rôle (AWS API)

1. (Facultatif) Si vous ne connaissez pas le nom du rôle que vous souhaitez modifier, appelez l'opération suivante pour répertorier les rôles de votre compte :

   - ListRoles
2. (Facultatif) Pour afficher la politique d'approbation actuelle d'un rôle, appelez l'opération suivante :

   - GetRole
3. Pour modifier les principaux approuvés ayant accès au rôle, créez un fichier texte avec la politique d'approbation mise à jour. Vous pouvez utiliser l'éditeur de texte pour créer la politique.

   Par exemple, la politique d'approbation suivante illustre comment référencer deux Comptes AWS dans l'élément `Principal`. Cela permet à des utilisateurs séparés Comptes AWS de jouer ce rôle.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {"AWS": [
            "arn:aws:iam::111122223333:root",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "sts:AssumeRole"
    }
}
```

Si vous spécifiez un principal dans un autre compte, l'ajout d'un compte à la politique d'approbation d'un rôle n'est qu'une partie de la procédure d'établissement de la relation d'approbation entre comptes. Par défaut, aucun utilisateur des comptes approuvés ne peut endosser le rôle. L'administrateur du compte nouvellement approuvé doit accorder aux utilisateurs l'autorisation d'endosser le rôle. Pour ce faire, l'administrateur doit créer ou modifier une politique attachée à l'utilisateur pour lui permettre d'accéder à l'action `sts:AssumeRole`. Pour plus d'informations, consultez la procédure suivante ou Octroi d'autorisations à un utilisateur pour endosser un rôle.

4. Pour utiliser le fichier créé afin de mettre à jour la politique de confiance, appelez l'opération suivante :

   - UpdateAssumeRolePolicy

Pour autoriser les utilisateurs d'un compte externe approuvé à utiliser le rôle (AWS API)

Pour plus d'informations sur cette procédure, consultez Octroi d'autorisations à un utilisateur pour endosser un rôle.

1. Créez un fichier JSON contenant une politique d'autorisations qui accorde des autorisations permettant d'endosser le rôle. Par exemple, la politique suivante contient les autorisations nécessaires minimales :

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": {
           "Effect": "Allow",
           "Action": "sts:AssumeRole",
           "Resource": "arn:aws:iam::111122223333:role/ROLE-NAME"
       }
   }
   ```

   Remplacez l'ARN dans l'instruction par celui du rôle que l'utilisateur peut endosser.

2. Appelez l'opération suivante pour charger le fichier JSON contenant la politique de confiance dans IAM :

- CreatePolicy

Le résultat de cette opération inclut l'ARN de la politique. Notez cet ARN, car vous aurez besoin de l'utiliser ultérieurement.

3. Décidez à quel utilisateur ou groupe attacher la politique. Si vous ne connaissez pas le nom de l'utilisateur ou du groupe concerné, appelez les opérations suivantes pour répertorier les utilisateurs ou les groupes de votre compte :

   - ListUsers
   - ListGroups

4. Appelez l'une des opérations suivantes pour attacher la politique créée à l'étape précédente à l'utilisateur ou au groupe :

   - API : AttachUserPolicy
   - AttachGroupPolicy

Mettre à jour les autorisations pour un rôle

Utilisez les procédures suivantes pour mettre à jour les politiques d'autorisation et les limites des autorisations d'un rôle.

Prérequis : afficher l'accès au rôle

Avant de modifier les autorisations d'un rôle, vous devez passer en revue ses activités récentes au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Mettre à jour une politique d'autorisation pour un rôle

Pour modifier les autorisations accordées par le rôle, modifiez la stratégie des autorisations du rôle (ou les stratégies). Vous ne pouvez pas modifier la politique d'autorisations d'un rôle lié à un service dans IAM. Vous pouvez modifier la politique d'autorisations dans le service dépendant du rôle. Pour vérifier si un service prend en charge cette fonctionnalité, reportez-vous à la rubrique AWS services qui fonctionnent avec IAM et recherchez les services qui possèdent Yes (Oui)dans la colonne Rôles

liés à un service. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Mise à jour d'une politique d'autorisations de rôle (console)

Pour changer les autorisations autorisées par un rôle (console)

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console IAM, choisissez Rôles.
3. Choisissez le nom du rôle à modifier, puis choisissez l'onglet Autorisations.
4. Effectuez l'une des actions suivantes :

   - Pour modifier une politique existante gérée par le client, choisissez le nom de la politique, puis choisissez Modifier la politique.

     > **Note**
     >
     > Vous ne pouvez pas modifier une politique AWS gérée.
     > AWS les politiques gérées apparaissent avec l' AWS icône
     > ().
     > Pour plus d'informations sur la différence entre les politiques AWS gérées et les politiques gérées par le client, consultezPolitiques gérées et politiques en ligne.

   - Pour attacher une stratégie gérée existante au rôle, choisissez Add permissions (Ajouter des autorisations) puis choisissez Attach policies (Attacher des politiques).
   - Pour modifier une politique en ligne existante, développez la politique et choisissez Edit (Modifier).
   - Pour intégrer une nouvelle politique en ligne, choisissez Add permissions (Ajouter des autorisations) puis choisissez Create inline policy (Création de stratégie en ligne).
   - Pour supprimer une politique existante du rôle, cochez la case située en regard du nom de la politique et sélectionnez Retirer.

Mise à jour d'une politique d'autorisations de rôle (AWS CLI)

Pour modifier les autorisations accordées par le rôle, modifiez la stratégie des autorisations du rôle (ou les stratégies). Vous ne pouvez pas modifier la politique d'autorisations d'un rôle lié à un service dans IAM. Vous pouvez modifier la politique d'autorisations dans le service dépendant du rôle. Pour

vérifier si un service prend en charge cette fonctionnalité, reportez-vous à la rubrique AWS services qui fonctionnent avec IAM et recherchez les services qui possèdent Yes (Oui)dans la colonne Rôles liés à un service. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Pour modifier les autorisations autorisées par un rôle (AWS CLI)

1. (Facultatif) Pour afficher les autorisations actuelles associées à un rôle, exécutez les commandes suivantes :

   1. aws vise à list-role-policies répertorier les politiques en ligne
   2. aws iam va list-attached-role-policies répertorier les politiques gérées

2. La commande permettant de mettre à jour les autorisations du rôle varie selon que vous mettez à jour une politique gérée ou une politique en ligne.

   Pour mettre à jour une politique gérée, exécutez la commande suivante pour créer une nouvelle version de la politique gérée :

   - était un objectif create-policy-version

   Pour mettre à jour une politique en ligne, exécutez la commande suivante :

   - était un objectif put-role-policy

Mettre à jour une politique d'autorisation de rôle (AWS API)

Pour modifier les autorisations accordées par le rôle, modifiez la stratégie des autorisations du rôle (ou les stratégies). Vous ne pouvez pas modifier la politique d'autorisations d'un rôle lié à un service dans IAM. Vous pouvez modifier la politique d'autorisations dans le service dépendant du rôle. Pour vérifier si un service prend en charge cette fonctionnalité, reportez-vous à la rubrique AWS services qui fonctionnent avec IAM et recherchez les services qui possèdent Yes (Oui)dans la colonne Rôles liés à un service. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Pour modifier les autorisations accordées par un rôle (AWS API)

1. (Facultatif) Pour afficher les autorisations actuelles associées à un rôle, appelez les opérations suivantes :

1. ListRolePoliciespour répertorier les politiques en ligne
2. ListAttachedRolePoliciespour répertorier les politiques gérées

2. L'opération permettant de mettre à jour les autorisations du rôle varie selon que vous mettez à jour une politique gérée ou une politique en ligne.

   Pour mettre à jour une politique gérée, appelez l'opération suivante pour créer une nouvelle version de la politique gérée :

   - CreatePolicyVersion

   Pour mettre à jour une politique en ligne, appelez l'opération suivante :

   - PutRolePolicy

Mettre à jour la limite des autorisations pour un rôle

Pour modifier le nombre maximum d'autorisations permises pour un rôle, modifiez la limite d'autorisations du rôle

Mise à jour d'une limite des autorisations de rôle (console)

Pour modifier la politique utilisée afin de définir la limite d'autorisations pour un rôle

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Rôles.
3. Sélectionnez le nom du rôle présentant la limite d'autorisations que vous souhaitez modifier.
4. Sélectionnez l'onglet Autorisations. Si nécessaire, ouvrez la section Permissions boundary (Limite d'autorisations), puis choisissez Change boundary (Modifier une limite).
5. Sélectionnez la politique que vous souhaitez utiliser pour la limite d'autorisations.
6. Choisissez Change boundary (Modifier une limite).

   Vos modifications ne prennent pas effet jusqu'à ce qu'une autre personne endosse ce rôle.

Mise à jour d’une limite des autorisations de rôle (AWS CLI)

Pour modifier la politique gérée utilisée afin de définir la limite d'autorisations pour un rôle (AWS CLI)

1. (Facultatif) Pour afficher la limite d'autorisations actuelle d'un rôle, exécutez la commande suivante :

   - aws iam get-role

2. Pour utiliser une autre politique gérée afin de mettre à jour la limite d'autorisations d'un rôle, exécutez la commande suivante :

   - était un objectif put-role-permissions-boundary

   Un rôle peut disposer d'une seule politique gérée définie comme une limite d'autorisations. Si vous modifiez la limite d'autorisations, cela modifie le nombre maximum d'autorisations permises pour un rôle.

Mettre à jour la limite d'autorisations d'un rôle (AWS API)

Pour modifier la politique gérée utilisée pour définir la limite des autorisations pour un rôle (AWS API)

1. (Facultatif) Pour afficher la limite d'autorisations actuelle d'un rôle, appelez l'opération suivante :

   - GetRole

2. Pour utiliser une autre politique gérée afin de mettre à jour la limite d'autorisations d'un rôle, appelez l'opération suivante :

   - PutRolePermissionsBoundary

   Un rôle peut disposer d'une seule politique gérée définie comme une limite d'autorisations. Si vous modifiez la limite d'autorisations, cela modifie le nombre maximum d'autorisations permises pour un rôle.

Mise à jour des paramètres pour un rôle

Utilisez les procédures suivantes pour mettre à jour la description d’un rôle ou modifier la durée maximale de session d’un rôle.

Mettre à jour la description d'un rôle

Pour modifier la description du rôle, modifiez le texte de la description.

Mise à jour d'une description de rôle (console)

Pour modifier la description d'un rôle (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de la console IAM, choisissez Rôles.
3. Choisissez le nom du rôle à modifier.
4. Dans la section Summary (Récapitulatif), sélectionnez Edit (Modifier).
5. Saisissez une nouvelle description dans le champ et sélectionnez Save changes (Enregistrer les modifications).

Mise à jour de la description d'un rôle (AWS CLI)

Pour modifier la description d'un rôle (AWS CLI)

1. (Facultatif) Pour afficher la description actuelle d'un rôle, exécutez la commande suivante :
   - aws iam get-role
2. Pour mettre à jour la description d'un rôle, exécutez la commande suivante avec le paramètre de description :
   - aws iam update-role

Mettre à jour la description d'un rôle (AWS API)

Pour modifier la description d'un rôle (AWS API)

1. (Facultatif) Pour afficher la description actuelle d'un rôle, appelez l'opération suivante :
   - GetRole
2. Pour mettre à jour la description d'un rôle, appelez l'opération suivante avec le paramètre de description :
   - UpdateRole

Mettre à jour la durée de session maximale pour un rôle

Pour spécifier le paramètre de durée de session maximale pour les rôles assumés à l'aide de la console, de l' AWS CLI AWS API ou de l'API, modifiez la valeur du paramètre de durée maximale de session. La valeur de ce paramètre peut varier de 1 heure à 12 heures. Si vous ne spécifiez pas de valeur, la valeur par défaut de 1 heure maximum est appliquée. Ce paramètre ne limite pas les sessions prises en charge par AWS les services.

Mise à jour de la durée de session maximale d’un rôle (console)

Pour modifier le paramètre de durée maximale de session pour les rôles assumés à l'aide de la console ou de AWS l'API (console) AWS CLI

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de la console IAM, choisissez Rôles.
3. Choisissez le nom du rôle à modifier.
4. Dans la section Summary (Récapitulatif), sélectionnez Edit (Modifier).
5. Pour Maximum session duration (Durée de session maximale), choisissez une valeur. Autrement, vous pouvez choisir Custom duration (Durée personnalisée) et saisir une valeur (en secondes).
6. Sélectionnez Enregistrer les modifications.

   Vos modifications ne prennent pas effet jusqu'à ce qu'une autre personne endosse ce rôle. Pour apprendre à révoquer des sessions existantes pour ce rôle, consultez Révocation des informations d’identification de sécurité temporaires d’un rôle IAM.

Dans le Console de gestion AWS, les sessions utilisateur IAM durent 12 heures par défaut. Les utilisateurs IAM qui changent de rôle dans la console se voient accorder la durée de session maximale du rôle ou le temps restant dans la session de l'utilisateur IAM, selon la durée la plus courte.

Toute personne assumant le rôle depuis l' AWS API AWS CLI or peut demander une session plus longue, jusqu'à ce maximum. Le paramètre `MaxSessionDuration` détermine la durée maximale de la session de rôle qui peut être demandée.

- Pour spécifier une durée de session à l'aide du paramètre, AWS CLI utilisez le `duration-seconds` paramètre. Pour en savoir plus, veuillez consulter la section Passez à un rôle IAM (AWS CLI).
- Pour spécifier une durée de session à l'aide de l' AWS API, utilisez le `DurationSeconds` paramètre. Pour en savoir plus, veuillez consulter la section Passez à un rôle IAM (AWS API).

Mise à jour de la durée de session maximale d'un rôle (AWS CLI)

> **Note**
>
> Toute personne assumant le rôle depuis l'API AWS CLI or peut utiliser le paramètre `duration-seconds` CLI ou le paramètre `DurationSeconds` API pour demander une session plus longue. Le paramètre `MaxSessionDuration` détermine la durée maximale de la session de rôle qui peut être demandée à l'aide du paramètre `DurationSeconds`. Si les utilisateurs ne spécifient pas de valeur pour le paramètre `DurationSeconds`, leurs informations d'identification de sécurité sont valides pendant une heure.

Pour modifier le paramètre de durée maximale de session pour les rôles assumés à l'aide de AWS CLI (AWS CLI)

1. (Facultatif) Pour afficher le paramètre de durée de session maximale d'un rôle, exécutez la commande suivante :

   - aws iam get-role

2. Pour mettre à jour le paramètre de durée de session maximale d'un rôle, exécutez la commande suivante avec le paramètre `max-session-duration` de l'interface de ligne de commande ou le paramètre `MaxSessionDuration` de l'API :

   - aws iam update-role

   Vos modifications ne prennent pas effet jusqu'à ce qu'une autre personne endosse ce rôle. Pour apprendre à révoquer des sessions existantes pour ce rôle, consultez Révocation des informations d'identification de sécurité temporaires d'un rôle IAM.

Mise à jour de la durée maximale des sessions de rôle (AWS API)

> **Note**
>
> Toute personne assumant le rôle depuis l'API AWS CLI or peut utiliser le paramètre `duration-seconds` CLI ou le paramètre `DurationSeconds` API pour demander une session plus longue. Le paramètre `MaxSessionDuration` détermine la durée maximale de la session de rôle qui peut être demandée à l'aide du paramètre `DurationSeconds`. Si les utilisateurs ne spécifient pas de valeur pour le paramètre `DurationSeconds`, leurs informations d'identification de sécurité sont valides pendant une heure.

Pour modifier le paramètre de durée maximale de session pour les rôles assumés à l'aide de l'API (AWS API)

1. (Facultatif) Pour afficher le paramètre de durée de session maximale d'un rôle, appelez l'opération suivante :

   - GetRole

2. Pour mettre à jour le paramètre de durée de session maximale d'un rôle, appelez l'opération suivante avec le paramètre `max-session-duration` de la CLI ou le paramètre `MaxSessionDuration` de l'API :

   - UpdateRole

   Vos modifications ne prennent pas effet jusqu'à ce qu'une autre personne endosse ce rôle. Pour apprendre à révoquer des sessions existantes pour ce rôle, consultez Révocation des informations d'identification de sécurité temporaires d'un rôle IAM.

Suppression de rôles ou de profils d'instance

Si vous n'avez plus besoin d'un rôle, nous vous recommandons de supprimer le rôle et ses autorisations associées. De cette façon, vous n'avez aucune entité inutilisée qui n'est pas surveillée ou gérée activement.

Si le rôle était associé à une instance EC2, vous pouvez également le supprimer du profil d'instance, puis supprimer ce dernier.

> **Warning**
>
> Vérifiez qu'aucune instance Amazon EC2 n'est en cours d'exécution avec le rôle ou le profil d'instance que vous êtes sur le point de supprimer. La suppression d'un rôle ou d'un profil d'instance associé à une instance en cours d'exécution arrêtera toutes les applications qui s'exécutent sur l'instance.

Si vous préférez ne pas supprimer définitivement un rôle, vous pouvez le désactiver. Pour ce faire, modifiez les politiques du rôle, puis révoquez toutes les sessions en cours. Par exemple, vous pouvez ajouter une politique au rôle qui refuse l'accès à tous AWS. Vous pouvez également modifier la politique d'approbation pour refuser l'accès à toute personne qui tente d'endosser le rôle. Pour en savoir plus sur la révocation des sessions, consultez Révocation des informations d’identification de sécurité temporaires d’un rôle IAM.

Rubriques

- Affichage de l’accès au rôle
- Suppression d'un rôle lié à un service
- Suppression d'un rôle IAM (console)
- Suppression d'un rôle IAM (AWS CLI)
- Supprimer un rôle IAM (AWS API)
- Informations connexes

Affichage de l’accès au rôle

Avant de supprimer un rôle, nous vous recommandons de vérifier la date de sa dernière utilisation. Vous pouvez le faire à l'aide de l'API Console de gestion AWS AWS CLI, du ou de l' AWS API. Vous devriez consulter ces informations, car vous ne voulez pas retirer l'accès à une personne utilisant ce rôle.

La date de la dernière activité du rôle peut ne pas correspondre à la dernière date indiquée dans l’onglet Dernière consultation. L’onglet Dernière consultation signale l’activité uniquement pour les services autorisés par les politiques d’autorisations du rôle. La date de la dernière activité du rôle inclut la dernière tentative d'accès à un service dans lequel le rôle a été créé AWS.

> **Note**
>
> La période de suivi de la dernière activité d'un rôle et des données Dernière consultation correspond aux 400 jours précédents. Cette période peut être plus courte si votre région a commencé à prendre en charge ces fonctionnalités au cours de la dernière année. Le rôle aurait pu être utilisé il y a plus de 400 jours. Pour de plus amples informations sur la période de suivi, veuillez consulter Où AWS suit les dernières informations consultées.

Pour afficher la date à laquelle un rôle a été utilisé pour la dernière fois (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Rôles.
3. Recherchez la ligne du rôle dont vous souhaitez afficher l'activité. Vous pouvez utiliser le champ de recherche pour affiner les résultats. Affichez la colonne Dernière activité pour voir le nombre de jours écoulés depuis la dernière utilisation du rôle. Si le rôle n'a pas été utilisé pendant la période de suivi, le tableau affiche Aucun.
4. Choisissez le nom du rôle pour afficher plus d'informations. La page Summary (Résumé) du rôle inclut également Last activity (Dernière activité), qui affiche la date à laquelle le rôle a été utilisé pour la dernière fois. Si le rôle n'a pas été utilisé au cours des 400 derniers jours, la Dernière activité affiche Non consulté pendant la période de suivi.

Pour voir quand un rôle a été utilisé pour la dernière fois (AWS CLI)

`aws iam get-role` - Exécutez cette commande pour renvoyer des informations sur un rôle, y compris l'objet `RoleLastUsed`. Cet objet contient la `LastUsedDate` et la `Region` dans laquelle le rôle a été utilisé pour la dernière fois. Si `RoleLastUsed` est présent mais ne contient pas de valeur, le rôle n'a pas été utilisé pendant la période de suivi.

Pour voir quand un rôle a été utilisé pour la dernière fois (AWS API)

`GetRole` - Appelez cette opération pour renvoyer des informations sur un rôle, y compris l'objet `RoleLastUsed`. Cet objet contient la `LastUsedDate` et la `Region` dans laquelle le rôle a été utilisé pour la dernière fois. Si `RoleLastUsed` est présent mais ne contient pas de valeur, le rôle n'a pas été utilisé pendant la période de suivi.

Suppression d'un rôle lié à un service

La méthode que vous utilisez pour supprimer un rôle lié à un service dépend dudit service. Dans certains cas, vous n'avez pas besoin de supprimer manuellement un rôle lié à un service. Par exemple, lorsque vous terminez une action donnée (comme supprimer une ressource) dans le service, le service peut supprimer le rôle lié au service à votre place. Dans d'autres cas, le service peut prendre en charge la suppression manuelle d'un rôle lié à un service à partir de la console, de l'API ou de la AWS CLI.

Consultez la documentation relative au rôle lié au service dans le service lié pour savoir comment supprimer le rôle. Pour afficher les rôles liés à un service de votre compte, rendez-vous sur la page IAM Roles (Rôles IAM) de la console. Les rôles liés à un service s'affichent avec (Rôle lié à un service) mentionné dans la colonne Entités de confiance du tableau. Une bannière sur la page Summary (Résumé) d'un rôle indique aussi que le rôle est lié à un service.

Si le service ne contient pas de documentation pour supprimer le rôle lié au service, vous pouvez utiliser la console IAM ou l'API pour supprimer le rôle. AWS CLI

Suppression d'un rôle IAM (console)

Lorsque vous utilisez le Console de gestion AWS pour supprimer un rôle, IAM détache automatiquement les politiques gérées associées au rôle. Il supprime aussi automatiquement toute politique en ligne associée au rôle, ainsi que tout profil d'instance Amazon EC2 qui contient le rôle.

> **Important**
>
> Dans certains cas, un rôle peut être associé à un profil d'instance Amazon EC2, et le rôle et le profil d'instance peuvent porter exactement le même nom. Dans ce cas, vous pouvez utiliser le Console de gestion AWS pour supprimer le rôle et le profil d'instance. Ce lient se fait automatiquement pour les rôles et les profils d'instance que vous créez dans la console. Si vous avez créé le rôle à partir des outils pour Windows PowerShell ou de l' AWS API, le rôle et le profil d'instance peuvent porter des noms différents. AWS CLI Dans ce cas, vous ne pouvez pas utiliser la console pour les supprimer. Vous devez plutôt utiliser les AWS CLI outils pour Windows PowerShell ou l' AWS API pour supprimer d'abord le rôle du profil d'instance. Vous devez ensuite réaliser une étape distincte pour supprimer le rôle.

Pour supprimer un rôle (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Rôles, puis cochez la case en regard du rôle à supprimer.
3. En haut de la page, sélectionnez Delete (Supprimer).
4. Dans la boîte de dialogue de confirmation, passez en revue les dernières informations consultées, qui indiquent la date à laquelle chacun des rôles sélectionnés a accédé pour la dernière fois à un AWS service. Cela vous permet de confirmer si le rôle est actif actuellement. Pour poursuivre, saisissez le nom du rôle dans le champ de saisie de texte et choisissez Delete (Supprimer). Si vous êtes sûr, procédez à la suppression même si les dernières informations consultées sont toujours en cours de chargement.

> **Note**
>
> Vous ne pouvez pas utiliser la console pour supprimer un profil d'instance, sauf lorsqu'elle porte le même nom que le rôle. Le profil d'instance est supprimé dans le cadre du processus de suppression d'un rôle comme décrit dans la procédure précédente. Pour supprimer un profil d'instance sans supprimer également le rôle, vous devez utiliser l' AWS API AWS CLI or. Pour plus d'informations, consultez les sections suivantes.

Suppression d'un rôle IAM (AWS CLI)

Lorsque vous utilisez le AWS CLI pour supprimer un rôle, vous devez d'abord supprimer les politiques intégrées associées au rôle. Vous devez également détacher les politiques gérées associées au rôle. Si vous voulez supprimer le profil d'instance associé contenant le rôle, vous devez le supprimer séparément.

Pour supprimer un rôle (AWS CLI)

1. Si vous ne connaissez pas le nom du rôle que vous souhaitez supprimer, tapez la commande suivante pour répertorier les rôles dans votre compte :

```
aws iam list-roles
```

La liste inclut l'Amazon Resource Name (ARN) de chaque rôle. Utilisez le nom du rôle, pas l'ARN, pour faire référence aux rôles avec les commandes CLI. Par exemple, si un rôle a l'ARN : `arn:aws:iam::123456789012:role/myrole`, vous faites référence au rôle en tant que **`myrole`**.

2. Supprimez le rôle de tous les profils d'instance auxquels il est associé.

   a. Pour répertorier tous les profils d'instance auxquels un rôle est associé, tapez la commande suivante :

   ```
   aws iam list-instance-profiles-for-role --role-name role-name
   ```

   b. Pour supprimer le rôle d'un profil d'instance, saisissez la commande suivante pour chaque profil instance :

   ```
   aws iam remove-role-from-instance-profile --instance-profile-name instance-profile-name --role-name role-name
   ```

3. Supprimez toutes les politiques associées au rôle.

   a. Pour répertorier toutes les politiques en ligne que contient le rôle, saisissez la commande suivante :

   ```
   aws iam list-role-policies --role-name role-name
   ```

   b. Pour supprimer chaque politique en ligne du rôle, saisissez la commande suivante pour chaque politique :

   ```
   aws iam delete-role-policy --role-name role-name --policy-name policy-name
   ```

   c. Pour répertorier toutes les politiques gérées attachées au rôle, saisissez la commande suivante :

   ```
   aws iam list-attached-role-policies --role-name role-name
   ```

   d. Pour détacher chaque politique gérée du rôle, saisissez la commande suivante pour chaque politique :

   ```
   aws iam detach-role-policy --role-name role-name --policy-arn policy-arn
   ```

4. Tapez la commande suivante pour supprimer le rôle :

```
aws iam delete-role --role-name role-name
```

5. Si vous ne prévoyez pas de réutiliser les profils d'instance associés au rôle, vous pouvez saisir la commande suivante pour les supprimer :

```
aws iam delete-instance-profile --instance-profile-name instance-profile-name
```

Supprimer un rôle IAM (AWS API)

Lorsque vous utilisez l'API IAM pour supprimer un rôle, vous devez d'abord supprimer les politiques en ligne associées au rôle. Vous devez également détacher les politiques gérées associées au rôle. Si vous voulez supprimer le profil d'instance associé contenant le rôle, vous devez le supprimer séparément.

Pour supprimer un rôle (AWS API)

1. Pour répertorier tous les profils d'instance auxquels un rôle est associé, appelez ListInstanceProfilesForRole.

   Pour supprimer le rôle d'un profil d'instance, appelez RemoveRoleFromInstanceProfile. Vous devez transmettre le nom du rôle et le nom du profil d'instance.

   Si vous ne comptez pas réutiliser un profil d'instance associé au rôle, appelez DeleteInstanceProfilepour le supprimer.
2. Pour répertorier toutes les politiques intégrées à un rôle, appelez ListRolePolicies.

   Pour supprimer les politiques intégrées associées au rôle, appelez DeleteRolePolicy. Vous devez transmettre le nom du rôle et le nom de la politique en ligne.
3. Pour répertorier toutes les politiques gérées associées à un rôle, appelez ListAttachedRolePolicies.

   Pour détacher les politiques gérées associées au rôle, appelez DetachRolePolicy. Vous devez transmettre le nom du rôle et l'ARN de la politique gérée.
4. Appelez DeleteRolepour supprimer le rôle.

Informations connexes

Pour obtenir des informations générales sur les profils d'instance, consultez Utilisation des profils d'instance.

Pour obtenir des informations générales sur les rôles liés à un service, veuillez consulter Créer un rôle lié à un service.

Méthodes pour assumer un rôle

Avant qu'un utilisateur, une application ou un service soit en mesure d'utiliser un rôle que vous avez créé, vous devez lui accorder les autorisations nécessaires pour changer de rôle. Pour accorder ces autorisations, vous pouvez utiliser n'importe quelle politique attachée à l'un des groupes ou des utilisateurs. Une fois les autorisations accordées, l'utilisateur peut assumer un rôle dans les outils pour Windows PowerShell, le AWS Command Line Interface (AWS CLI) et l'`AssumeRole`API. Console de gestion AWS

> **Important**
>
> Si vous créez un rôle par programmation plutôt que dans la console IAM, vous avez la possibilité d'ajouter un chemin `Path` de 512 caractères au maximum à l'élément `RoleName` qui lui, est de 64 caractères au maximum. Toutefois, si vous avez l'intention d'utiliser un rôle avec la fonctionnalité Switch Role dans le Console de gestion AWS, la combinaison `Path` `RoleName` ne peut pas dépasser 64 caractères.

La méthode utilisée pour endosser le rôle détermine qui peut endosser le rôle et la durée de la session de rôle. Lorsque vous l'utilisez `AssumeRole*` les opérations d'API, le rôle IAM que vous endossez est la ressource. Le rôle ou l'utilisateur IAM qui appelle les opérations de l'API `AssumeRole*` est le principal.

Le tableau suivant compare les méthodes pour endosser des rôles.

Méthode permettant d'endosser le rôle	Qui peut endosser le rôle	Méthode permettant de spécifier la durée de vie des informations d'identification	Durée de vie des informations d'identification (min | max | par défaut)
Console de gestion AWS	Utilisateur ou rôles[1] (en changeant de rôle)	Durée de session maximale sur la page récapitulative des rôles	15 min | Durée de session maximale[2] | 1 h
Opération d'interface de ligne de commande (CLI) `assume-role` ou d'API `AssumeRole`	Utilisateur ou rôle[1]	Paramètre de CLI `duration-seconds` ou d'API `DurationSeconds`	15 min | Durée de session maximale[2] | 1 h
Opération d'interface de ligne de commande (CLI) `assume-role-with-saml` ou d'API `AssumeRoleWithSAML`	Tout utilisateur authentifié utilisant SAML	Paramètre de CLI `duration-seconds` ou d'API `DurationSeconds`	15 min | Durée de session maximale[2] | 1 h
Opération d'interface de ligne de commande (CLI) `assume-role-with-w`	Tout utilisateur authentifié à l'aide d'un fournisseur OIDC	Paramètre de CLI `duration-seconds` ou d'API `DurationSeconds`	15 min | Durée de session maximale[2] | 1 h

Méthode permettant d'endosser le rôle	Qui peut endosser le rôle	Méthode permettant de spécifier la durée de vie des informations d'identification	Durée de vie des informations d'identification (min | max | par défaut)
`eb-identity` ou d'API `AssumeRoleWithWebIdentity`			
URL de console construite avec `AssumeRole`	Utilisateur ou rôle	Paramètre HTML `SessionDuration` dans l'URL	15 min | 12 h | 1 h
URL de console construite avec `AssumeRoleWithSAML`	Tout utilisateur authentifié utilisant SAML	Paramètre HTML `SessionDuration` dans l'URL	15 min | 12 h | 1 h
URL de console construite avec `AssumeRoleWithWebIdentity`	Tout utilisateur authentifié à l'aide d'un fournisseur OIDC	Paramètre HTML `SessionDuration` dans l'URL	15 min | 12 h | 1 h

[1] L'utilisation des informations d'identification pour qu'un rôle endosse un rôle différent est appelée création de chaînes de rôles. Lorsque vous utilisez la création de chaînes de rôles, la durée de session du rôle est limitée à une heure. Cela s'applique au changement de Console de gestion AWS rôle et aux opérations d'API. AWS CLI Cette limitation ne s'applique pas à l'attribution initiale d'un rôle à partir des informations d'identification de l'utilisateur, ni aux applications exécutées sur des EC2 instances Amazon à l'aide de profils d'instance.

[2] La valeur de ce paramètre peut varier de 1 heure à 12 heures. Pour en savoir plus sur la modification du paramètre de durée de session maximale, consultez Gestion des rôles IAM. Ce paramètre détermine la durée de session maximale que vous pouvez demander lorsque vous obtenez les informations d'identification du rôle. Par exemple, lorsque vous utilisez les opérations d'API AssumeRole* pour assumer un rôle, vous pouvez spécifier une durée de session à l'aide du `DurationSeconds` paramètre. Utilisez ce paramètre pour spécifier la durée de la session du rôle entre 900 secondes (15 minutes) et la durée de session maximale pour le rôle. Les utilisateurs IAM qui changent de rôle dans la console se voient accorder la durée de session maximale ou le temps restant dans la session de l'utilisateur, selon la durée la plus courte. Supposons que vous définissiez une durée maximale de 5 heures sur un rôle. Un utilisateur IAM qui s'est connecté à la console pendant 10 heures (sur 12, le maximum par défaut) décide d'endosser le rôle. La durée de la session de rôle disponible est de 2 heures. Pour savoir comment afficher la valeur maximale pour votre rôle, consultez Mettre à jour la durée de session maximale pour un rôle plus loin sur cette page.

Remarques

- Le paramètre de durée maximale de session ne limite pas les sessions endossées par les services AWS .
- Les informations d'identification du rôle Amazon EC2 IAM ne sont pas soumises à la durée de session maximale configurée dans le rôle.
- Pour permettre aux utilisateurs d'assumer à nouveau le rôle actuel au cours d'une session de rôle, spécifiez l'ARN ou l' Compte AWS ARN comme principal dans la politique de confiance des rôles. Services AWS qui fournissent des ressources de calcul telles qu'Amazon EC2, Amazon ECS, Amazon EKS et Lambda fournissent des informations d'identification temporaires et mettent automatiquement à jour ces informations d'identification. Cela garantit que vous disposez toujours d'un ensemble d'informations d'identification valide. Pour ces services, il n'est pas nécessaire d'endosser à nouveau le rôle actuel pour obtenir des informations d'identification temporaires. Toutefois, si vous avez l'intention de transférer des balises de session ou une politique de session, vous devez endosser à nouveau le rôle actuel. Pour savoir comment modifier une politique d'approbation des rôles afin d'ajouter l'ARN ou Compte AWS l'ARN du rôle principal, consultezMise à jour d'une politique d'approbation de rôle.

Rubriques

- Basculer d'un utilisateur à un rôle IAM (console)

- Passez à un rôle IAM (AWS CLI)
- Passer à un rôle IAM (Outils pour Windows PowerShell)
- Passez à un rôle IAM (AWS API)
- Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2
- Utilisation des profils d'instance

Basculer d'un utilisateur à un rôle IAM (console)

Vous pouvez changer de rôle lorsque vous vous connectez en tant qu'utilisateur IAM, utilisateur dans IAM Identity Center, rôle ou SAML-federated rôle fédéré d'identité Web. Un rôle spécifie un ensemble d'autorisations que vous pouvez utiliser pour accéder aux AWS ressources dont vous avez besoin. Toutefois, vous ne vous connectez pas au rôle mais, une fois connecté en tant qu'utilisateur IAM, vous pouvez passer à un rôle IAM. Ceci annule temporairement vos autorisations utilisateur d'origine et les remplace par celles affectées au rôle. Le rôle peut être dans votre propre compte ou dans un autre Compte AWS. Pour plus d'informations sur les rôles, leurs avantages et la façon de les créer, consultez Rôles IAM et Création d'un rôle IAM.

Les autorisations de votre utilisateur et des rôles que vous assumez ne peuvent pas être cumulées. Un seul ensemble d'autorisations peut être actif à la fois. Lorsque vous endossez un rôle, vous abandonnez temporairement vos autorisations utilisateur et travaillez avec celles qui sont affectées au rôle. Lorsque vous quittez le rôle, vos autorisations utilisateur sont automatiquement restaurées.

Lorsque vous changez de rôle dans le Console de gestion AWS, la console utilise toujours vos informations d'identification d'origine pour autoriser le changement. Par exemple, si vous basculez sur RoleA, IAM utilise vos informations d'identification d'origine pour déterminer si vous êtes autorisé à endosser le rôle RoleA. Si vous passez ensuite au rôle B alors que vous utilisez le rôle A, AWS il utilise toujours vos informations d'identification d'origine pour autoriser le changement, et non les informations d'identification pour le rôle A.

> **Note**
>
> Lorsque vous vous connectez en tant qu'utilisateur dans IAM Identity Center, en tant que SAML-federated rôle ou en tant que rôle fédéré d'identité Web, vous assumez un rôle IAM lorsque vous démarrez votre session. Par exemple, lorsqu'un utilisateur d'IAM Identity Center se connecte au portail d' AWS accès, il peut choisir un ensemble d'autorisations ou accéder

à un rôle attribué via le gestionnaire d'accès au compte avant de pouvoir accéder aux AWS ressources.

Séances de rôle

Lorsque vous changez de rôle, votre Console de gestion AWS session dure 1 heure par défaut. Les sessions des utilisateurs IAM ont une durée de 12 heures par défaut, d'autres utilisateurs peuvent avoir des durées de session différentes. Lorsque vous changez de rôle dans la console, la durée maximale de la session du rôle ou la durée restante de votre session d'utilisateur vous est accordée, selon la durée la plus courte. Vous ne pouvez pas prolonger la durée de votre session en endossant un rôle.

Par exemple, supposons qu'une durée de session maximale de 10 heures soit définie pour un rôle. Vous êtes connecté à la console depuis huit heures lorsque vous décidez d'endosser le rôle. Il reste quatre heures dans votre session utilisateur IAM, de sorte que la durée autorisée de la session du rôle est de quatre heures, et non la durée maximale de la session de dix heures. Le tableau suivant montre comment déterminer la durée de session d'un utilisateur IAM lorsqu'il change de rôle dans la console.

Le temps restant de la session de l'utilisateur IAM est...	La durée de la session de rôle est…		
Moins que la durée de session maximale d'un rôle	Temps restant dans la session de l'utilisateur IAM		
Plus que la durée de session maximale du rôle	Valeur de durée de session maximale		
Égale à la durée de session maximale du rôle	Valeur de durée de session maximale (approximative)		

L'utilisation des informations d'identification pour qu'un rôle endosse un rôle différent est appelée création de chaînes de rôles. Lorsque vous utilisez le chaînage des rôles, la durée de la session est limitée à une heure, quel que soit le paramètre de durée maximale de session configuré pour chaque rôle. Cela s'applique au changement de Console de gestion AWS rôle et aux opérations d'API. AWS CLI

> **Note**
>
> Certaines consoles AWS de service peuvent renouveler automatiquement votre session de rôle lorsqu'elle expire sans que vous preniez aucune mesure. Certaines peuvent vous inviter à recharger la page de votre navigateur pour réauthentifier votre session.

Considérations

- Vous ne pouvez pas changer de rôle si vous vous connectez en tant que Utilisateur racine d'un compte AWS.
- Les utilisateurs doivent être autorisés à changer de rôle conformément à la politique. Pour obtenir des instructions, veuillez consulter Octroi d'autorisations à un utilisateur pour endosser un rôle.
- Vous ne pouvez pas passer d'un rôle Console de gestion AWS à un rôle qui nécessite une ExternalId valeur. Vous ne pouvez basculer vers un tel rôle qu'en appelant l'API `AssumeRole` qui prend en charge le paramètre `ExternalId`.

Pour changer de rôle

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à Console de gestion AWS du Guide de l'utilisateur Connexion à AWS .
2. Dans le Console de gestion AWS, choisissez votre nom d'utilisateur dans la barre de navigation en haut à droite. Cela ressemble généralement à ceci : ***username*** @ ***account_ID_number_or_alias***.
3. Choisissez l'une des méthodes suivantes pour changer de rôle :
   - Choisissez Changer de rôle.
   - Si vous avez opté pour la prise en charge multi-session, choisissez Ajouter une session et sélectionnez Changer de rôle.

> **Note**
>
> Vous pouvez vous connecter à un maximum de cinq identités différentes simultanément dans un seul navigateur Web dans l' Console de gestion AWS. Pour plus de détails, consultez la section Connexion à plusieurs comptes dans le Guide de démarrage Console de gestion AWS .

4. Dans la page Switch Role (Changer de rôle), entrez l'ID ou l'alias de compte, ainsi que le nom du rôle que vous a fourni l'administrateur.

> **Note**
>
> Si l'administrateur a créé le rôle avec un chemin d'accès tel que `division_abc/subdivision_efg/roleToDoX`, vous devez entrer le chemin d'accès complet et le nom dans le champ Rôle. Si vous entrez uniquement le nom du rôle, ou si l'ensemble de `Path` et `RoleName` dépasse 64 caractères, le changement de rôle échoue. Cette restriction est imposée par les cookies du navigateur dans lesquels est stocké le nom du rôle. Dans ce cas, contactez votre administrateur et demandez-lui de réduire la taille du chemin d'accès et le nom du rôle.

5. (Facultatif) Vous pouvez saisir un nom d'affichage et sélectionner une couleur d'affichage qui mettra en évidence le rôle dans la barre de navigation de la console.

   - Pour Nom d'affichage, saisissez le texte que vous voulez afficher sur la barre de navigation à la place de votre nom utilisateur lorsque le rôle est actif. Un nom, basé sur les informations du compte et du rôle, est suggéré, mais vous pouvez le modifier à votre convenance.
   - Pour Couleur d'affichage, sélectionnez une couleur pour mettre en évidence le nom d'affichage.

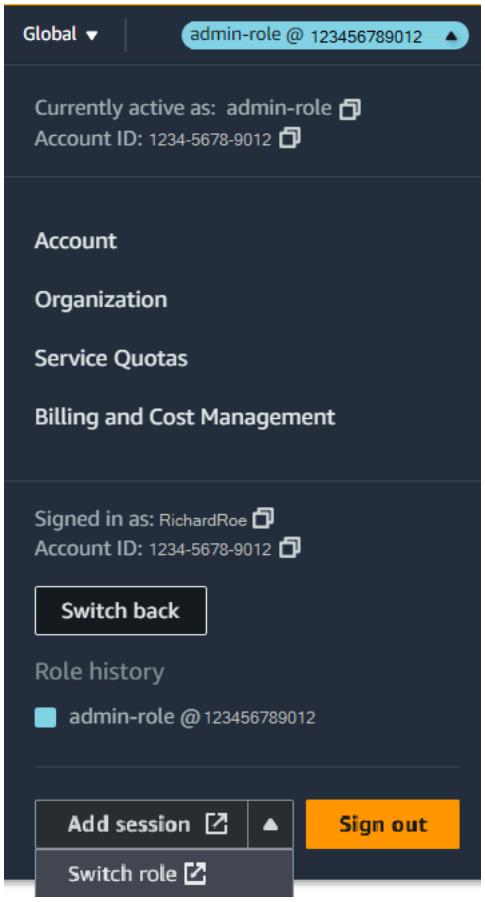
   Le nom et la couleur peuvent vous aider à savoir quand le rôle est actif, ce qui modifie vos autorisations. Par exemple, pour un rôle qui vous donne accès à l'environnement de test, vous pouvez spécifier un Display name (Nom d'affichage) de **`Test`** et sélectionner la couleur verte pour Color. Pour le rôle qui vous donne accès à la production, vous pouvez spécifier un Display name (Nom d'affichage) de **`Production`** et sélectionner la couleur rouge pour Color.

6. Choisissez Changer de rôle. Le nom d'affichage et la couleur remplacent votre nom utilisateur sur la barre de navigation et vous pouvez commencer à utiliser les autorisations que le rôle vous accorde.
7. Une fois que vous avez terminé les tâches nécessitant le rôle IAM, vous pouvez revenir à votre session d'origine. Cela supprimera les autorisations supplémentaires fournies par le rôle et vous ramènera à vos autorisations standard.

   a. Dans la console IAM, choisissez le nom d'affichage du rôle sous Display Name (Nom d'affichage) en haut à droite de la barre de navigation.

   b. Choisissez Revenir.

   Par exemple, supposons que vous êtes connecté au numéro de compte `123456789012` à l'aide du nom d'utilisateur `Richard`. Une fois que vous avez utilisé le rôle `admin-role`, vous souhaitez cesser d'utiliser ce rôle et revenir à vos autorisations initiales. Pour arrêter d'utiliser le rôle, sélectionnez admin-role @ 123456789012, puis choisissez Switch back.



> **Tip**
>
> Les derniers rôles utilisés sont répertoriés dans le menu . La prochaine fois que vous voulez endosser l'un de ces rôles, choisissez simplement le rôle souhaité. Il vous suffit de saisir manuellement les informations sur le compte et le rôle si le rôle n'est pas affiché dans le menu.

Ressources supplémentaires

- Octroi d'autorisations à un utilisateur pour endosser un rôle
- Accorder à un utilisateur l'autorisation de transmettre un rôle à un AWS web
- Créer un rôle pour attribuer des autorisations à un utilisateur IAM
- Créez un rôle pour déléguer des autorisations à un AWS web
- Résolution des problèmes liés aux rôles IAM

Passez à un rôle IAM (AWS CLI)

Un rôle spécifie un ensemble d'autorisations que vous pouvez utiliser pour accéder aux ressources AWS dont vous avez besoin. À cet égard, il est semblable à un utilisateur IAM Gestion des identités et des accès AWS. Lorsque vous vous connectez en tant qu'utilisateur, un ensemble spécifique d'autorisations vous est affecté. Toutefois, vous ne vous connectez pas au rôle mais, une fois connecté en tant qu'utilisateur, vous pouvez passer à un rôle. Ceci annule temporairement vos autorisations utilisateur d'origine et les remplace par celles affectées au rôle. Le rôle peut être dans votre propre compte ou dans un autre Compte AWS. Pour plus d'informations sur les rôles, leurs avantages et la façon de les créer et de les configurer, consultez Rôles IAM et Création d'un rôle IAM. Pour connaître les différentes méthodes que vous pouvez utiliser pour endosser un rôle, consultez Méthodes pour assumer un rôle.

> **Important**
>
> Les autorisations de votre utilisateur IAM et des rôles que vous endossez ne peuvent pas être cumulées. Un seul ensemble d'autorisations peut être actif à la fois. Lorsque vous endossez un rôle, vous abandonnez temporairement les autorisations utilisateur et de rôle précédentes et utilisez celles qui sont affectées au rôle. Lorsque vous quittez le rôle, vos autorisations utilisateur sont automatiquement restaurées.

Vous pouvez utiliser un rôle pour exécuter une AWS CLI commande lorsque vous êtes connecté en tant qu'utilisateur IAM. Vous pouvez également utiliser un rôle pour exécuter une AWS CLI commande lorsque vous êtes connecté en tant qu'utilisateur authentifié en externe (SAML ou OIDC) qui utilise déjà un rôle. De plus, vous pouvez utiliser un rôle pour exécuter une commande de la AWS CLI dans une instance Amazon EC2 attachée à un rôle via son profil d'instance. Il n'est pas possible d'assumer un rôle si vous êtes connecté en tant qu' Utilisateur racine d'un compte AWS.

Chaînage de rôles : vous pouvez aussi utiliser le chaînage de rôles, qui utilise les autorisations d'un rôle pour accéder à un second rôle.

Par défaut, votre session de rôle dure une heure. Lorsque vous endossez ce rôle à l'aide des opérations de la CLI `assume-role*`, vous pouvez spécifier une valeur pour le paramètre `duration-seconds`. Cette valeur peut être comprise entre 900 secondes (15 minutes) et la valeur de durée de session maximale définie pour le rôle. Si vous changez de rôle dans la console, la durée de votre session est limitée à une heure. Pour savoir comment afficher la valeur maximale pour votre rôle, veuillez consulter Mettre à jour la durée de session maximale pour un rôle.

Si vous utilisez la création de chaînes de rôles, la durée de votre session est limitée à une heure maximum. Si vous utilisez ensuite le paramètre `duration-seconds` pour fournir une valeur supérieure à une heure, l'opération échoue.

Exemple de scénario : passer à un rôle de production

Imaginez que vous êtes un utilisateur IAM pour travailler dans l'environnement de développement. Dans ce scénario, vous devez parfois travailler avec l'environnement de production au niveau de la ligne de commande avec l'AWS CLI. Vous disposez déjà d'un ensemble d'informations d'identification de clé d'accès. Il peut s'agir de la paire de clés d'accès attribuée à votre utilisateur IAM standard. Ou, si vous êtes connecté en tant que principal fédéré SAML ou OIDC, il peut s'agir de la paire de clés d'accès pour le rôle qui vous a été attribué initialement. Si vos autorisations actuelles vous permettent d'assumer un rôle IAM spécifique, vous pouvez identifier ce rôle dans un « profil » des fichiers de AWS CLI configuration. Cette commande est ensuite exécutée avec les autorisations du rôle IAM spécifié, pas l'identité d'origine. Notez que lorsque vous spécifiez ce profil dans une AWS CLI commande, vous utilisez le nouveau rôle. Dans ce cas, vous ne pouvez pas utiliser vos autorisations d'origine dans le compte de développement en même temps. Ceci est dû au fait qu'un seul ensemble d'autorisations peut être actif à la fois.

> **Note**
>
> Pour des raisons de sécurité, les administrateurs peuvent consulter AWS CloudTrail les journaux pour savoir qui a effectué une action AWS. Votre administrateur peut exiger que vous spécifiiez une identité source ou un nom de session de rôle lorsque vous endossez le rôle. Pour plus d'informations, consultez `sts:SourceIdentity` et `sts:RoleSessionName`.

Pour passer à un rôle de production (AWS CLI)

1. Si vous n'avez jamais utilisé le AWS CLI, vous devez d'abord configurer votre profil CLI par défaut. Ouvrez une invite de commande et configurez votre AWS CLI installation pour utiliser la clé d'accès de votre utilisateur IAM ou de votre rôle fédéré. Pour plus d'informations, veuillez consulter configuration de l'outil AWS Command Line Interface dans le guide de l'utilisateur de l'outil AWS Command Line Interface .

   Exécutez la commande aws configure comme suit :

   ```
   aws configure
   ```

   Lorsque vous y êtes invité, fournissez les informations suivantes :

   ```
   AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
   AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
   Default region name [None]: us-east-2
   Default output format [None]: json
   ```

2. Créez un profil pour le rôle dans le `.aws/config` fichier Unix ou Linux, ou le fichier `C:\Users\USERNAME\.aws\config` dans Windows. L'exemple suivant crée un profil appelé `prodaccess` qui passe au rôle *`ProductionAccessRole`* dans le compte `123456789012`. Vous obtenez l'ARN du rôle auprès de l'administrateur du compte ayant créé le rôle. Lorsque ce profil est invoqué, il AWS CLI utilise les informations d'identification du `source_profile` pour demander des informations d'identification pour le rôle. De ce fait, l'identité référencée en tant que `source_profile` doit disposer des autorisations `sts:AssumeRole` pour le rôle spécifié dans le `role_arn`.

   ```
   [profile prodaccess]
       role_arn = arn:aws:iam::123456789012:role/ProductionAccessRole
   ```

```
    source_profile = default
```

3. Une fois que vous avez créé le nouveau profil, toute AWS CLI commande qui spécifie le paramètre `--profile prodaccess` s'exécute avec les autorisations associées `ProductionAccessRole` au rôle IAM au lieu de l'utilisateur par défaut.

   ```
   aws iam list-users --profile prodaccess
   ```

   Cette commande fonctionne si les autorisations affectées au rôle `ProductionAccessRole` permettent de répertorier les utilisateurs dans le compte AWS actuel.
4. Pour revenir aux autorisations accordées par vos informations d'identification d'origine, exécutez les commandes sans le paramètre `--profile`. Vous revenez AWS CLI à l'utilisation des informations d'identification de votre profil par défaut, dans Step 1 lequel vous avez configuré.

Pour plus d'informations, consultez Endossement d'un rôle dans le Guide de l'utilisateur AWS Command Line Interface .

Exemple de scénario : permettre à un rôle de profil d'instance de passer à un rôle dans un autre compte

Imaginez que vous en utilisiez deux Comptes AWS et que vous souhaitiez autoriser une application exécutée sur une instance Amazon EC2 à exécuter des AWS CLI commandes sur les deux comptes. Supposons que l'instance EC2 existe dans le compte `111111111111`. Cette instance inclut le rôle de profil d'instance `abcd` qui permet à l'appli d'effectuer les tâches Amazon S3 `amzn-s3-demo-bucket1` en lecture seule sur le compartiment dans le même compte `111111111111`. Toutefois, l'application doit également être autorisée à endosser le rôle `efgh` entre comptes pour effectuer des tâches dans le compte `222222222222`. Pour ce faire, le rôle de profil d'instance EC2 `abcd` doit disposer des autorisations suivantes :

Politique d'autorisations de rôle ***abcd*** du compte 111111111111

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
```

```
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1/*",
                "arn:aws:s3:::amzn-s3-demo-bucket1"
            ]
        },
        {
            "Sid": "AllowIPToAssumeCrossAccountRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

Supposons que le rôle `efgh` entre comptes permet aux tâches Amazon S3 en lecture seule sur le compartiment `amzn-s3-demo-bucket2` dans le même compte `222222222222`. Pour ce faire, le rôle `efgh` entre comptes doit avoir la politique d'autorisations suivante :

Politique d'autorisations de rôle ***efgh*** du compte 222222222222

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket2/*",
                "arn:aws:s3:::amzn-s3-demo-bucket2"
            ]
        }
    ]
}
```

Le rôle `efgh` doit autoriser le rôle de profil d'instance `abcd` à l'endosser. Pour ce faire, le rôle `efgh` doit avoir la politique de confiance suivante :

Politique de confiance de rôle ***efgh*** du compte 222222222222

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
```

```
    ]
}
```

Pour exécuter ensuite AWS CLI les commandes dans le compte222222222222, vous devez mettre à jour le fichier de configuration de la CLI. Identifiez le rôle `efgh` en tant que « profil » et le rôle de profil d'instance EC2 `abcd` en tant que « source » d'informations d'identification dans le fichier de configuration d' AWS CLI . Ensuite, vos commandes de CLI sont exécutées avec les autorisations du rôle `efgh`, pas du rôle `abcd` d'origine.

> **Note**
>
> Pour des raisons de sécurité, vous pouvez l'utiliser AWS CloudTrail pour auditer l'utilisation des rôles dans le compte. Pour différencier les sessions de rôle lorsqu'un rôle est utilisé par différents acteurs principaux dans les CloudTrail journaux, vous pouvez utiliser le nom de la session de rôle. Lorsque le AWS CLI assume un rôle pour le compte d'un utilisateur, comme décrit dans cette rubrique, un nom de session de rôle est automatiquement créé sous la forme`AWS-CLI-session-`*nnnnnnnn*. *nnnnnnnn*Voici un entier qui représente l'heure en temps d'époque Unix (le nombre de secondes écoulées depuis minuit UTC le 1er janvier 1970). Pour plus d'informations, consultez la section Référence des CloudTrail événements dans le Guide de AWS CloudTrail l'utilisateur.

Pour permettre à un rôle de profil d'instance EC2 de passer à un rôle multicompte (AWS CLI)

1. Vous n'avez pas besoin de configurer un profil par défaut pour la CLI. Au lieu de cela, vous pouvez charger les informations d'identification à partir des métadonnées du profil d'instance EC2. Créez un profil pour le rôle dans le fichier `.aws/config`. L'exemple suivant crée un profil `instancecrossaccount` qui passe au rôle *`efgh`* dans le compte `222222222222`. Lorsque ce profil est appelé, l'interface AWS CLI utilise les informations d'identification des métadonnées du profil d'instance EC2 pour demander les informations d'identification du rôle. De ce fait, le rôle de profil d'instance EC2 doit disposer des autorisations `sts:AssumeRole` pour le rôle spécifié dans le `role_arn`.

   ```
   [profile instancecrossaccount]
   role_arn = arn:aws:iam::222222222222:role/efgh
   credential_source = Ec2InstanceMetadata
   ```

2. Une fois que vous avez créé le nouveau profil, toute AWS CLI commande qui spécifie le paramètre `--profile instancecrossaccount` s'exécute avec les autorisations associées au `efgh` rôle dans le compte`222222222222`.

   ```
   aws s3 ls amzn-s3-demo-bucket2 --profile instancecrossaccount
   ```

   Cette commande fonctionne si les autorisations affectées au rôle `efgh` autorisent à répertorier les utilisateurs dans l' Compte AWS actuel.

3. Pour revenir au profil d'instance EC2 d'origine les autorisations de compte `111111111111`, exécutez les commandes de CLI sans le paramètre `--profile`.

Pour plus d'informations, consultez Endossement d'un rôle dans le Guide de l'utilisateur AWS Command Line Interface .

Passer à un rôle IAM (Outils pour Windows PowerShell)

Un rôle spécifie un ensemble d'autorisations que vous pouvez utiliser pour accéder aux ressources AWS dont vous avez besoin. À cet égard, il est semblable à un utilisateur IAM Gestion des identités et des accès AWS. Lorsque vous vous connectez en tant qu'utilisateur, un ensemble spécifique d'autorisations vous est affecté. Toutefois, vous ne vous connectez pas au rôle mais, une fois connecté, vous pouvez endosser un rôle. Ceci annule temporairement vos autorisations utilisateur d'origine et les remplace par celles affectées au rôle. Le rôle peut être dans votre propre compte ou dans un autre Compte AWS. Pour plus d'informations sur les rôles, leurs avantages et la façon de les créer et de les configurer, consultez Rôles IAM et Création d'un rôle IAM.

> **Important**
>
> Les autorisations de votre utilisateur IAM et des rôles que vous endossez ne peuvent pas être cumulées. Un seul ensemble d'autorisations peut être actif à la fois. Lorsque vous endossez un rôle, vous abandonnez temporairement vos autorisations utilisateur et travaillez avec celles qui sont affectées au rôle. Lorsque vous quittez le rôle, vos autorisations utilisateur sont automatiquement restaurées.

Cette section décrit comment changer de rôles lorsque vous utilisez la ligne de commande avec AWS Tools for Windows PowerShell.

Imaginez que vous possédez un compte dans l'environnement de développement et que vous ayez parfois besoin de travailler avec l'environnement de production en ligne de commande à l'aide des Outils pour Windows PowerShell. Un ensemble d'informations d'identification de clé d'accès est déjà à votre disposition. Il peut s'agir de la paire de clés d'accès attribuée à votre utilisateur IAM standard. Ou, si vous êtes connecté en tant que principal fédéré SAML ou OIDC, il peut s'agir de la paire de clés d'accès pour le rôle qui vous a été attribué initialement. Ces informations d'identification vous permettent d'exécuter le cmdlet `Use-STSRole` qui transmet l'ARN d'un nouveau rôle en tant que paramètre. Cette commande renvoie es informations d'identification de sécurité temporaires du rôle demandé. Vous pouvez ensuite utiliser ces informations d'identification dans PowerShell les commandes suivantes avec les autorisations du rôle pour accéder aux ressources en production. Bien que vous utilisiez ce rôle, vous ne pouvez pas utiliser vos autorisations d'utilisateur dans le compte Développement, car un seul ensemble d'autorisations est en vigueur à la fois.

> **Note**
>
> Pour des raisons de sécurité, les administrateurs peuvent consulter AWS CloudTrail les journaux pour savoir qui a effectué une action AWS. Votre administrateur peut exiger que vous spécifiiez une identité source ou un nom de session de rôle lorsque vous endossez le rôle. Pour plus d'informations, consultez `sts:SourceIdentity` et `sts:RoleSessionName`.

Notez que toutes les clés d'accès et tous les jetons sont des exemples uniquement et ne peuvent pas être utilisés comme indiqué. Remplacez-les par les valeurs correspondantes de votre environnement en direct.

Pour passer à un rôle (Outils pour Windows PowerShell)

1. Ouvrez une invite de PowerShell commande et configurez le profil par défaut pour utiliser la clé d'accès de votre utilisateur IAM actuel ou de votre rôle fédéré. Si vous avez déjà utilisé les Outils pour Windows PowerShell, cela est probablement déjà fait. Notez que vous ne pouvez changer de rôle que si vous êtes connecté en tant qu'utilisateur IAM, et non en tant que Utilisateur racine d'un compte AWS.

```
PS C:\> Set-AWSCredentials -AccessKey AKIAIOSFODNN7EXAMPLE -
SecretKey wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY -StoreAs MyMainUserProfile
PS C:\> Initialize-AWSDefaults -ProfileName MyMainUserProfile -Region us-east-2
```

Pour plus d'informations, consultez la section Utilisation des AWS informations d'identification dans le Guide de Outils AWS pour PowerShell l'utilisateur.

2. Pour récupérer les informations d'identification du nouveau rôle, exécutez la commande suivante pour passer au rôle *RoleName* dans le compte 123456789012. Vous obtenez l'ARN du rôle auprès de l'administrateur du compte ayant créé le rôle. La commande nécessite que vous fournissiez un nom de session également. Pour cela, n'importe quel texte fera l'affaire. La commande suivante demande les informations d'identification et capture l'objet de propriété `Credentials` dans l'objet des résultats renvoyés et le stocke dans la variable `$Creds`.

```
PS C:\> $Creds = (Use-STSRole -RoleArn "arn:aws:iam::123456789012:role/RoleName" -
RoleSessionName "MyRoleSessionName").Credentials
```

`$Creds` est un objet qui contient à présent les éléments `AccessKeyId`, `SecretAccessKey` et `SessionToken` dont vous avez besoin dans la procédure suivante. Les exemples de commandes suivants illustrent les valeurs habituelles.

```
PS C:\> $Creds.AccessKeyId
AKIAIOSFODNN7EXAMPLE

PS C:\> $Creds.SecretAccessKey
wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

PS C:\> $Creds.SessionToken
AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLECvSRyh0FW7jEXAMPLEW+vE/7s1HRp
XviG7b+qYf4nD00EXAMPLEmj4wxS04L/uZEXAMPLECihzFB5lTYLto9dyBgSDyEXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPy
Oj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UuysgsKdEXAMPLE1TVastU1A0SKFEXAMPLEiyw
C
s8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP+4eZScEXAMPLEsnf87eNhyDHq6ikBQ==

PS C:\> $Creds.Expiration
Thursday, June 18, 2018 2:28:31 PM
```

3. Pour utiliser ces informations d'identification dans n'importe quelle commande suivante, incluez-les dans le paramètre `-Credential`. Par exemple, la commande suivante utilise les informations d'identification du rôle et fonctionne uniquement si le rôle a l'autorisation `iam:ListRoles` et peut donc exécuter le cmdlet `Get-IAMRoles` :

```
PS C:\> get-iamroles -Credential $Creds
```

4. Pour revenir à vos informations d'identification d'origine, arrêtez simplement d'utiliser le `-Credentials $Creds` paramètre et PowerShell autorisez le retour aux informations d'identification stockées dans le profil par défaut.

Passez à un rôle IAM (AWS API)

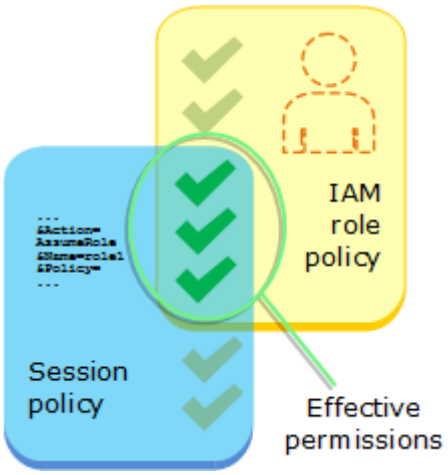
Un rôle spécifie un ensemble d'autorisations que vous pouvez utiliser pour accéder aux ressources AWS . À cet égard, il est semblable à un utilisateur IAM. Un principal (personne ou application) assume le rôle de recevoir des autorisations temporaires pour effectuer les tâches requises et interagir avec les AWS ressources. Le rôle peut être dans votre propre compte ou dans un autre Compte AWS. Pour plus d'informations sur les rôles, leurs avantages et la façon de les créer et de les configurer, consultez Rôles IAM et Création d'un rôle IAM. Pour connaître les différentes méthodes que vous pouvez utiliser pour endosser un rôle, consultez Méthodes pour assumer un rôle.

> **Important**
>
> Les autorisations de votre utilisateur IAM et des rôles que vous endossez ne peuvent pas être cumulées. Un seul ensemble d'autorisations peut être actif à la fois. Lorsque vous endossez un rôle, vous abandonnez temporairement les autorisations utilisateur et de rôle précédentes et utilisez celles qui sont affectées au rôle. Lorsque vous quittez le rôle, vos autorisations originales sont automatiquement restaurées.

Pour assumer un rôle, une application appelle l'opération AWS STS `AssumeRole`API et transmet l'ARN du rôle à utiliser. L'opération crée une nouvelle session avec des informations d'identification temporaires. Cette session possède les mêmes autorisations que les politiques basées sur une identité pour ce rôle.

Lorsque vous appelez `AssumeRole`, vous pouvez, si vous le souhaitez, transmettre des politiques de session en ligne ou gérées. Les politiques de session sont des politiques avancées que vous transmettez en tant que paramètre lorsque vous créez par programmation une session d'informations d'identification temporaires pour un rôle ou une session utilisateur fédérée. Vous pouvez transmettre un seul document de politique de session en ligne JSON à l'aide du paramètre `Policy`. Vous

pouvez utiliser le paramètre `PolicyArns` pour spécifier jusqu'à 10 politiques de session gérées. Les autorisations de la session obtenues sont une combinaison des stratégies basées sur l'identité de l'entité et des stratégies de session. Les politiques de session s'avèrent utiles lorsque vous devez fournir les informations d'identification temporaires du rôle à une autre personne. Cette dernière peut utiliser les informations d'identification temporaires du rôle dans les appels d'API AWS suivants pour accéder aux ressources du compte qui possède le rôle. Vous ne pouvez pas utiliser les politiques de session pour accorder plus d'autorisations que celles autorisées par la politique basée sur l'identité. Pour en savoir plus sur la manière de AWS déterminer les autorisations effectives d'un rôle, consultezLogique d'évaluation de politiques.



Vous pouvez appeler `AssumeRole` lorsque vous êtes connecté en tant qu'utilisateur IAM ou en tant qu'utilisateur authentifié en externe (SAML ou OIDC) utilisant déjà un rôle. Vous pouvez également utiliser la création de chaînes de rôles, qui utilise un rôle pour endosser un deuxième rôle. Il n'est pas possible d'assumer un rôle si vous êtes connecté en tant qu' Utilisateur racine d'un compte AWS.

Par défaut, votre session de rôle dure une heure. Lorsque vous assumez ce rôle à l'aide des opérations d' AWS STS `AssumeRole*`API, vous pouvez spécifier une valeur pour le `DurationSeconds` paramètre. Cette valeur peut être comprise entre 900 secondes (15 minutes) et la valeur de durée de session maximale définie pour le rôle. Pour savoir comment afficher la valeur maximale pour votre rôle, veuillez consulter Mettre à jour la durée de session maximale pour un rôle.

Si vous utilisez la création de chaînes de rôles, votre session est limitée à une durée maximale d'une heure. Si vous utilisez ensuite le paramètre `DurationSeconds` pour fournir une valeur supérieure à une heure, l'opération échoue.

> **Note**
>
> Pour des raisons de sécurité, les administrateurs peuvent consulter AWS CloudTrail les journaux pour savoir qui a effectué une action dans AWS. Votre administrateur peut exiger que vous spécifiiez une identité source ou un nom de session de rôle lorsque vous endossez le rôle. Pour plus d'informations, consultez `sts:SourceIdentity` et `sts:RoleSessionName`.

Les exemples de code suivants montrent comment créer un utilisateur et assumer un rôle.

> **Warning**
>
> Afin d'éviter les risques de sécurité, n'employez pas les utilisateurs IAM pour l'authentification lorsque vous développez des logiciels spécialisés ou lorsque vous travaillez avec des données réelles. Préférez la fédération avec un fournisseur d'identité tel que AWS IAM Identity Center.

- Créer un utilisateur sans autorisation.
- Créer un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3 pour le compte.
- Ajouter une politique pour permettre à l'utilisateur d'assumer le rôle.
- Assumez le rôle et répertorier les compartiments S3 à l'aide d'informations d'identification temporaires, puis nettoyez les ressources.

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
global using Amazon.IdentityManagement;
```

```
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
```

```
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }


    /// <summary>
```

```
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
```

```
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

```
    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
 { UserName = userName });
```

```
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
```

```
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
```

```
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
```

```
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }


    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }
```

```
        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
```

```
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
 policyName, string policyDocument)
```

```
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
 accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}
```

```
using Microsoft.Extensions.Configuration;

namespace IAMBasics;

public class IAMBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();


        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
```

```
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();

        // First create a user. By default, the new user has
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;

        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
              "\"Effect\": \"Allow\"," +
              "\"Principal\": {" +
              $" \"AWS\": \"{userArn}\"" +
              "}," +
              "\"Action\": \"sts:AssumeRole\"" +
          "}]" +
        "}";

        // Permissions to list all buckets.
        string policyDocument = "{" +
            "\"Version\": \"2012-10-17\"," +
            " \"Statement\" : [{" +
                " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
                " \"Effect\" : \"Allow\"," +
                " \"Resource\" : \"*\"" +
            "}]" +
        "}";

        // Create an AccessKey for the user.
        uiWrapper.DisplayTitle("Create access key");
        Console.WriteLine("Now let's create an access key for the new user.");
        var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);
```

```
        var accessKeyId = accessKey.AccessKeyId;
        var secretAccessKey = accessKey.SecretAccessKey;

        Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");

        Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
        var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

        // Now try listing the Amazon Simple Storage Service (Amazon S3)
        // buckets. This should fail at this point because the user doesn't
        // have permissions to perform this task.
        uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
        Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
        var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
        var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
        var buckets = await s3Wrapper.ListMyBucketsAsync();

        Console.WriteLine(buckets is null
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");

        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
```

```
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);

        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);

        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        // Wait for the new credentials to become valid.
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");

        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);

        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);

        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
```

```
                Console.WriteLine($"{bucket.BucketName} created:
 {bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();

        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
 complete.");
        Console.WriteLine("Please wait while we clean up the resources we
 created.");

        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);

        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);

        uiWrapper.PressEnter();

        Console.WriteLine("All done cleaning up our resources. Thank you for your
 patience.");
    }
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;
```

```
    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
```

```
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
{ BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
{ BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
```

```
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
 stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
 full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
 creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
```

```
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
```

```
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour .NET .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole

- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
```

```
 echo
 echo "This demo will create an IAM user, create an IAM role, and apply the role
to the user."
 echo_repeat "*" 88
 echo

 echo -n "Enter a name for a new IAM user: "
 get_input
 user_name=$get_input_result

 local user_arn
 user_arn=$(iam_create_user -u "$user_name")

 # shellcheck disable=SC2181
 if [[ ${?} == 0 ]]; then
   echo "Created demo IAM user named $user_name"
 else
   errecho "$user_arn"
   errecho "The user failed to create. This demo will exit."
   return 1
 fi

 local access_key_response
 access_key_response=$(iam_create_user_access_key -u "$user_name")
 # shellcheck disable=SC2181
 if [[ ${?} != 0 ]]; then
   errecho "The access key failed to create. This demo will exit."
   clean_up "$user_name"
   return 1
 fi

 IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
 local key_name=${access_key_values[0]}
 local key_secret=${access_key_values[1]}

 echo "Created access key named $key_name"

 echo "Wait 10 seconds for the user to be ready."
 sleep 10
 echo_repeat "*" 88
 echo

 local iam_role_name
 iam_role_name=$(generate_random_name "test-role")
```

```
 echo "Creating a role named $iam_role_name with user $user_name as the
principal."

 local assume_role_policy_document="{
   \"Version\": \"2012-10-17\",
   \"Statement\": [{
       \"Effect\": \"Allow\",
       \"Principal\": {\"AWS\": \"$user_arn\"},
       \"Action\": \"sts:AssumeRole\"
       }]
   }"

 local role_arn
 role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Created IAM role named $iam_role_name"
 else
   errecho "The role failed to create. This demo will exit."
   clean_up "$user_name" "$key_name"
   return 1
 fi

 local policy_name
 policy_name=$(generate_random_name "test-policy")
 local policy_document="{
              \"Version\": \"2012-10-17\",
              \"Statement\": [{
                  \"Effect\": \"Allow\",
                  \"Action\": \"s3:ListAllMyBuckets\",
                  \"Resource\": \"arn:aws:s3:::*\"}]}"

 local policy_arn
 policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
 # shellcheck disable=SC2181
 if [[ ${?} == 0 ]]; then
   echo "Created  IAM policy named $policy_name"
 else
   errecho "The policy failed to create."
   clean_up "$user_name" "$key_name" "$iam_role_name"
   return 1
 fi
```

```
 if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
   echo "Attached policy $policy_arn to role $iam_role_name"
 else
   errecho "The policy failed to attach."
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
   return 1
 fi

 local assume_role_policy_document="{
               \"Version\": \"2012-10-17\",
               \"Statement\": [{
                   \"Effect\": \"Allow\",
                   \"Action\": \"sts:AssumeRole\",
                   \"Resource\": \"$role_arn\"}]}"

 local assume_role_policy_name
 assume_role_policy_name=$(generate_random_name "test-assume-role-")

 # shellcheck disable=SC2181
 local assume_role_policy_arn
 assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
 else
   errecho "The policy failed to create."
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
   return 1
 fi

 echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
 sleep 10
 echo_repeat "*" 88
 echo

 echo "Try to list buckets without the new user assuming the role."
 echo_repeat "*" 88
 echo

 # Set the environment variables for the created user.
```

```
 # bashsupport disable=BP2001
 export AWS_ACCESS_KEY_ID=$key_name
 # bashsupport disable=BP2001
 export AWS_SECRET_ACCESS_KEY=$key_secret

 local buckets
 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. This should not have
happened."
 else
   errecho "Because the role with permissions has not been assumed, listing
buckets failed."
 fi

 echo
 echo_repeat "*" 88
 echo "Now assume the role $iam_role_name and list the buckets."
 echo_repeat "*" 88
 echo

 local credentials

 credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Assumed role $iam_role_name"
 else
   errecho "Failed to assume role."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 IFS=$'\t ' read -r -a credentials <<<"$credentials"

 export AWS_ACCESS_KEY_ID=${credentials[0]}
 export AWS_SECRET_ACCESS_KEY=${credentials[1]}
```

```
  # bashsupport disable=BP2001
  export AWS_SESSION_TOKEN=${credentials[2]}

  buckets=$(s3_list_buckets)

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. Listing buckets
 succeeded because of "
    echo "the assumed role."
  else
    errecho "Failed to list buckets. This should not happen."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    export AWS_SESSION_TOKEN=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
 "$policy_arn" "$assume_role_policy_arn"
    return 1
  fi

  local result=0
  export AWS_ACCESS_KEY_ID=""
  export AWS_SECRET_ACCESS_KEY=""

  echo
  echo_repeat "*" 88
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

Fonctions IAM utilisées dans ce scénario.

```
##############################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
##############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

##############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
```

```
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
```

```
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
```

```
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
```

```
  echo "Creates an AWS Identity and Access Management (IAM) role."
  echo "  -n role_name   The name of the IAM role."
  echo "  -p policy_json -- The assume role policy document."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-role \
  --role-name "$role_name" \
  --assume-role-policy-document "$policy_document" \
  --output text \
  --query Role.Arn)

local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
```

```
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-policy \
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
```

```
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
```

```
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
```

```
 function usage() {
   echo "function iam_detach_role_policy"
   echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
role."
   echo "  -n role_name   The name of the IAM role."
   echo "  -p policy_ARN -- The IAM policy document ARN."
   echo ""
 }

 # Retrieve the calling parameters.
 while getopts "n:p:h" option; do
   case "${option}" in
     n) role_name="${OPTARG}" ;;
     p) policy_arn="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$role_name" ]]; then
   errecho "ERROR: You must provide a role name with the -n parameter."
   usage
   return 1
 fi

 if [[ -z "$policy_arn" ]]; then
   errecho "ERROR: You must provide a policy ARN with the -p parameter."
   usage
   return 1
 fi

 response=$(aws iam detach-role-policy \
   --role-name "$role_name" \
   --policy-arn "$policy_arn")

 local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an AWS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
```

```
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
```

```
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an AWS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""
```

```
  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an AWS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }
```

```
  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
```

```
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
```

```
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .

- AttachRolePolicy
- CreateAccessKey
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
namespace AwsDoc {
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
```

```
                                            const Aws::IAM::Model::Policy &policy);
    }

    static const int LIST_BUCKETS_WAIT_SEC = 20;

    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);

        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
 std::endl;
```

```
        }

        user = outcome.GetResult().GetUser();
    }

    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;

        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);

        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());
```

```
        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
    }

    // 3. Create an IAM policy.
    {
        Aws::IAM::Model::CreatePolicyRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String policyName = "iam-demo-policy-" +
                                 Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetPolicyName(policyName);

        // Build IAM policy document.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");
        jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
        jsonStatement.WithString("Resource", "arn:aws:s3:::*");
```

```
        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                  << std::endl;

        // Set IAM policy document as JSON string.
        request.SetPolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a policy with name, " <<
policyName <<
                      "." << std::endl;
        }

        policy = outcome.GetResult().GetPolicy();
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
```

```
        request.SetRoleSessionName(roleSessionName);

        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;

                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
        }

        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }


    // 5. List objects in the bucket (This should fail).
    {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
```

```
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if (listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
            }
            else {
                std::cout
                        << "Access to list buckets denied because privileges have
not been applied."
                        << std::endl;
            }
        }
        else {
            std::cerr
                    << "Successfully retrieved bucket lists when this should not
happen."
                    << std::endl;
        }
    }

    // 6. Attach the policy to the role.
    {
        Aws::IAM::Model::AttachRolePolicyRequest request;
        request.SetRoleName(role.GetRoleName());
        request.WithPolicyArn(policy.GetArn());

        Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
                request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
```

```
                    ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
    }

    int count = 0;
    // 7. List objects in the bucket (this should succeed).
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }

            std::this_thread::sleep_for(std::chrono::seconds(1));
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
                      << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
```

```
    return DeleteCreatedEntities(client, role, user, policy);
}

bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
    bool result = true;
    if (policy.ArnHasBeenSet()) {
        // Detach the policy from the role.
        {
            Aws::IAM::Model::DetachRolePolicyRequest request;
            request.SetPolicyArn(policy.GetArn());
            request.SetRoleName(role.GetRoleName());

            Aws::IAM::Model::DetachRolePolicyOutcome outcome =
 client.DetachRolePolicy(
                    request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error Detaching policy from roles. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully detached the policy with arn "
                          << policy.GetArn()
                          << " from role " << role.GetRoleName() << "." <<
 std::endl;
            }
        }

        // Delete the policy.
        {
            Aws::IAM::Model::DeletePolicyRequest request;
            request.WithPolicyArn(policy.GetArn());

            Aws::IAM::Model::DeletePolicyOutcome outcome =
 client.DeletePolicy(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error deleting policy. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
```

```
                std::cout << "Successfully deleted the policy with arn "
                          << policy.GetArn() << std::endl;
            }
        }

    }

    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour C++ .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Exécutez un scénario interactif à une invite de commande.

```
import (
 "context"
 "errors"
 "fmt"
 "log"
 "math/rand"
```

```
 "strings"
 "time"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/credentials"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/aws-sdk-go-v2/service/s3"
 "github.com/aws/aws-sdk-go-v2/service/sts"
 "github.com/aws/smithy-go"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/iam/actions"
)

// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
//  1. Create a user who has no permissions.
//  2. Create a role that grants permission to list Amazon Simple Storage Service
//     (Amazon S3) buckets for the account.
//  3. Add a policy to let the user assume the role.
//  4. Try and fail to list buckets without permissions.
//  5. Assume the role and list S3 buckets using temporary credentials.
//  6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
 sdkConfig      aws.Config
 accountWrapper actions.AccountWrapper
 policyWrapper  actions.PolicyWrapper
 roleWrapper    actions.RoleWrapper
 userWrapper    actions.UserWrapper
 questioner     demotools.IQuestioner
 helper         IScenarioHelper
 isTestRun      bool
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
```

```
 helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
 return AssumeRoleScenario{
  sdkConfig:      sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
  questioner:     questioner,
  helper:         helper,
 }
}

// addTestOptions appends the API options specified in the original configuration
 to
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
}

// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run(ctx context.Context) {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong with the demo.\n")
   log.Println(r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
 demo.")
 log.Println(strings.Repeat("-", 88))

 user := scenario.CreateUser(ctx)
 accessKey := scenario.CreateAccessKey(ctx, user)
 role := scenario.CreateRoleAndPolicies(ctx, user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(ctx, accessKey)
 scenario.ListBucketsWithAssumedRole(ctx, noPermsConfig, role)
```

```
 scenario.Cleanup(ctx, user, role)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}

// CreateUser creates a new IAM user. This user has no permissions.
func (scenario AssumeRoleScenario) CreateUser(ctx context.Context) *types.User {
 log.Println("Let's create an example user with no permissions.")
 userName := scenario.questioner.Ask("Enter a name for the example user:",
 demotools.NotEmpty{})
 user, err := scenario.userWrapper.GetUser(ctx, userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(ctx, userName)
  if err != nil {
   panic(err)
  }
  log.Printf("Created user %v.\n", *user.UserName)
 } else {
  log.Printf("User %v already exists.\n", *user.UserName)
 }
 log.Println(strings.Repeat("-", 88))
 return user
}

// CreateAccessKey creates an access key for the user.
func (scenario AssumeRoleScenario) CreateAccessKey(ctx context.Context, user
 *types.User) *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(ctx, *user.UserName)
 if err != nil {
  panic(err)
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return accessKey
}
```

```
// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(ctx context.Context,
 user *types.User) *types.Role {
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err := scenario.roleWrapper.CreateRole(ctx,
 scenario.helper.GetName(), *user.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  ctx, scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"},
 "arn:aws:s3:::*")
 if err != nil {
  panic(err)
 }
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
 err = scenario.roleWrapper.AttachRolePolicy(ctx, *listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {
  panic(err)
 }
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
 err = scenario.userWrapper.CreateUserPolicy(ctx, *user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
```

```
 return listBucketsRole
}

// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(ctx
 context.Context, accessKey *types.AccessKey) *aws.Config {
 log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 noPermsConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
  ))
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)

 s3Client := s3.NewFromConfig(noPermsConfig)
 _, err = s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
    panic(err)
   }
  }
 } else {
```

```
  log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
   "but the call succeeded. Continuing the example anyway...")
 }
 log.Println(strings.Repeat("-", 88))
 return &noPermsConfig
}

// ListBucketsWithAssumedRole performs the following actions:
//
//  1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//     the user's access key credentials.
//  2. Gets temporary credentials by assuming the role that grants permission to
 list the
//     buckets.
//  3. Creates an Amazon S3 client from the temporary credentials.
//  4. Lists buckets for the account. Because the temporary credentials are
 generated by
//     assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(ctx
 context.Context, noPermsConfig *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(ctx, &sts.AssumeRoleInput{
  RoleArn:         role.Arn,
  RoleSessionName: aws.String("AssumeRoleExampleSession"),
  DurationSeconds: aws.Int32(900),
 })
 if err != nil {
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
   *tempCredentials.Credentials.SecretAccessKey,
   *tempCredentials.Credentials.SessionToken),
  ),
 )
 if err != nil {
```

```
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&assumeRoleConfig)

 s3Client := s3.NewFromConfig(assumeRoleConfig)
 result, err := s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  log.Println("Couldn't list buckets with assumed role credentials.")
  panic(err)
 }
 log.Println("Successfully called ListBuckets with assumed role credentials, \n"
 +
  "here are some of them:")
 for i := 0; i < len(result.Buckets) && i < 5; i++ {
  log.Printf("\t%v\n", *result.Buckets[i].Name)
 }
 log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(ctx context.Context, user *types.User,
 role *types.Role) {
 if scenario.questioner.AskBool(
  "Do you want to delete the resources created for this example? (y/n)", "y",
 ) {
  policies, err := scenario.roleWrapper.ListAttachedRolePolicies(ctx,
 *role.RoleName)
  if err != nil {
   panic(err)
  }
  for _, policy := range policies {
   err = scenario.roleWrapper.DetachRolePolicy(ctx, *role.RoleName,
 *policy.PolicyArn)
   if err != nil {
    panic(err)
   }
   err = scenario.policyWrapper.DeletePolicy(ctx, *policy.PolicyArn)
   if err != nil {
    panic(err)
   }
   log.Printf("Detached policy %v from role %v and deleted the policy.\n",
```

```
    *policy.PolicyName, *role.RoleName)
  }
  err = scenario.roleWrapper.DeleteRole(ctx, *role.RoleName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted role %v.\n", *role.RoleName)

  userPols, err := scenario.userWrapper.ListUserPolicies(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  for _, userPol := range userPols {
   err = scenario.userWrapper.DeleteUserPolicy(ctx, *user.UserName, userPol)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
  }
  keys, err := scenario.userWrapper.ListAccessKeys(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  for _, key := range keys {
   err = scenario.userWrapper.DeleteAccessKey(ctx, *user.UserName,
 *key.AccessKeyId)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
 *user.UserName)
  }
  err = scenario.userWrapper.DeleteUser(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted user %v.\n", *user.UserName)
  log.Println(strings.Repeat("-", 88))
 }

}

// IScenarioHelper abstracts input and wait functions from a scenario so that
 they
```

```
// can be mocked for unit testing.
type IScenarioHelper interface {
 GetName() string
 Pause(secs int)
}

const rMax = 100000

type ScenarioHelper struct {
 Prefix string
 Random *rand.Rand
}

// GetName returns a unique name formed of a prefix and a random number.
func (helper *ScenarioHelper) GetName() string {
 return fmt.Sprintf("%v%v", helper.Prefix, helper.Random.Intn(rMax))
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
 time.Sleep(time.Duration(secs) * time.Second)
}
```

Définissez une structure qui encapsule les actions du compte.

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}
```

```
// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy(ctx context.Context)
 (*types.PasswordPolicy, error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(ctx,
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders(ctx context.Context)
 ([]types.SAMLProviderListEntry, error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(ctx,
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

Définissez une structure qui encapsule les actions de la politique.

```
import (
 "context"
 "encoding/json"
```

```
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(ctx context.Context, maxPolicies int32)
 ([]types.Policy, error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}



// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version   string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect    string
```

```
 Action    []string
 Principal map[string]string `json:",omitempty"`
 Resource  *string           `json:",omitempty"`
}

// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(ctx context.Context, policyName string,
 actions []string,
 resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(ctx, &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// GetPolicy gets data about a policy.
```

```
func (wrapper PolicyWrapper) GetPolicy(ctx context.Context, policyArn string)
 (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(ctx, &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(ctx context.Context, policyArn string)
 error {
 _, err := wrapper.IamClient.DeletePolicy(ctx, &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

Définissez une structure qui encapsule les actions du rôle.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)
```

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(ctx context.Context, maxRoles int32)
 ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(ctx,
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(ctx context.Context, roleName string,
 trustedUserArn string) (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:    "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action:    []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
```

```
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(ctx, &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(ctx context.Context, roleName string)
 (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(ctx,
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(ctx context.Context,
 serviceName string, description string) (
 *types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(ctx,
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
```

```
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(ctx context.Context, roleName
 string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(ctx,
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(ctx context.Context, policyArn
 string, roleName string) error {
 _, err := wrapper.IamClient.AttachRolePolicy(ctx, &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}
```

```
// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(ctx context.Context, roleName
 string) ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(ctx,
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(ctx context.Context, roleName string,
 policyArn string) error {
 _, err := wrapper.IamClient.DetachRolePolicy(ctx, &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(ctx context.Context, roleName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(ctx,
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
```

```
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(ctx context.Context, roleName string) error
 {
 _, err := wrapper.IamClient.DeleteRole(ctx, &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

Définissez une structure qui encapsule les actions de l'utilisateur.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)
```

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(ctx context.Context, maxUsers int32)
 ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(ctx, &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(ctx, &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
```

```
  user = result.User
 }
 return user, err
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(ctx, &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(ctx context.Context, userName string,
 policyName string, actions []string,
 roleArn string) error {
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
```

```
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(ctx, &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(ctx context.Context, userName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(ctx,
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(ctx context.Context, userName string,
 policyName string) error {
 _, err := wrapper.IamClient.DeleteUserPolicy(ctx, &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
```

```
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(ctx context.Context, userName string) error
 {
 _, err := wrapper.IamClient.DeleteUser(ctx, &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(ctx context.Context, userName
 string) (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(ctx, &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}



// DeleteAccessKey deletes an access key from a user.
```

```
func (wrapper UserWrapper) DeleteAccessKey(ctx context.Context, userName string,
 keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(ctx, &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(ctx context.Context, userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(ctx, &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour Go .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey

- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de l'utilisateur IAM.

```
/*
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
 */
```

```
public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static String userArn;

    public static void main(String[] args) throws Exception {

        final String usage = """

                Usage:
                    <username> <policyName> <roleName> <roleSessionName>
 <bucketName>\s

                Where:
                    username - The name of the IAM user to create.\s
                    policyName - The name of the policy to create.\s
                    roleName - The name of the role to create.\s
                    roleSessionName - The name of the session required for the
 assumeRole operation.\s
                    bucketName - The name of the Amazon S3 bucket from which
 objects are read.\s
                """;

        if (args.length != 5) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        String policyName = args[1];
        String roleName = args[2];
```

```
        String roleSessionName = args[3];
        String bucketName = args[4];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the AWS IAM example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
```

```
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
```

```
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());
```

```
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);

waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();
```

```
            ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
            String polArn;
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
this role.");
                    return;
                }
            }

            AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);
            System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();
```

```
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();

            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();

            System.out.println("Created a S3Client using temp credentials.");
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

```
   public static void deleteRole(IamClient iam, String roleName, String polArn)
{

       try {
           // First the policy needs to be detached.
           DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
                   .policyArn(polArn)
                   .roleName(roleName)
                   .build();

           iam.detachRolePolicy(rolePolicyRequest);

           // Delete the policy.
           DeletePolicyRequest request = DeletePolicyRequest.builder()
                   .policyArn(polArn)
                   .build();

           iam.deletePolicy(request);
           System.out.println("*** Successfully deleted " + polArn);

           // Delete the role.
           DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                   .roleName(roleName)
                   .build();

           iam.deleteRole(roleRequest);
           System.out.println("*** Successfully deleted " + roleName);

       } catch (IamException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   public static void deleteKey(IamClient iam, String username, String
accessKey) {
       try {
           DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                   .accessKeyId(accessKey)
                   .userName(username)
                   .build();

           iam.deleteAccessKey(request);
```

```
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK for Java 2.x .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy

- PutUserPolicy

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créer un utilisateur IAM et un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3. L'utilisateur n'a que le droit d'assumer le rôle. Après avoir assumé le rôle, utilisez des informations d'identification temporaires pour répertorier les compartiments pour le compte.

```
import {
  CreateUserCommand,
  GetUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import { ScenarioInput } from "@aws-doc-sdk-examples/lib/scenario/index.js";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "iam_basic_test_username";
const policyName = "iam_basic_test_policy";
const roleName = "iam_basic_test_role";
```

```
/**
 * Create a new IAM user. If the user already exists, give
 * the option to delete and re-create it.
 * @param {string} name
 */
export const createUser = async (name, confirmAll = false) => {
  try {
    const { User } = await iamClient.send(
      new GetUserCommand({ UserName: name }),
    );
    const input = new ScenarioInput(
      "deleteUser",
      "Do you want to delete and remake this user?",
      { type: "confirm" },
    );
    const deleteUser = await input.handle({}, { confirmAll });
    // If the user exists, and you want to delete it, delete the user
    // and then create it again.
    if (deleteUser) {
      await iamClient.send(new DeleteUserCommand({ UserName: User.UserName }));
      await iamClient.send(new CreateUserCommand({ UserName: name }));
    } else {
      console.warn(
        `${name} already exists. The scenario may not work as expected.`,
      );
      return User;
    }
  } catch (caught) {
    // If there is no user by that name, create one.
    if (caught instanceof Error && caught.name === "NoSuchEntityException") {
      const { User } = await iamClient.send(
        new CreateUserCommand({ UserName: name }),
      );
      return User;
    }
    throw caught;
  }
};

export const main = async (confirmAll = false) => {
  // Create a user. The user has no permissions by default.
  const User = await createUser(userName, confirmAll);
```

```
 if (!User) {
   throw new Error("User not created");
 }

 // Create an access key. This key is used to authenticate the new user to
 // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
(AWS STS).
 // It's not best practice to use access keys. For more information, see
https://aws.amazon.com/iam/resources/best-practices/.
 const createAccessKeyResponse = await iamClient.send(
   new CreateAccessKeyCommand({ UserName: userName }),
 );

 if (
   !createAccessKeyResponse.AccessKey?.AccessKeyId ||
   !createAccessKeyResponse.AccessKey?.SecretAccessKey
 ) {
   throw new Error("Access key not created");
 }

 const {
   AccessKey: { AccessKeyId, SecretAccessKey },
 } = createAccessKeyResponse;

 let s3Client = new S3Client({
   credentials: {
     accessKeyId: AccessKeyId,
     secretAccessKey: SecretAccessKey,
   },
 });

 // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
 // thrown while the user and access keys are still stabilizing.
 await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
   try {
     return await listBuckets(s3Client);
   } catch (err) {
     if (err instanceof Error && err.name === "InvalidAccessKeyId") {
       throw err;
     }
   }
 });
```

```
 // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
 // is thrown while the user and access keys are still stabilizing.
 const { Role } = await retry(
   {
     intervalInMs: 2000,
     maxRetries: 60,
   },
   () =>
     iamClient.send(
       new CreateRoleCommand({
         AssumeRolePolicyDocument: JSON.stringify({
           Version: "2012-10-17",
           Statement: [
             {
               Effect: "Allow",
               Principal: {
                 // Allow the previously created user to assume this role.
                 AWS: User.Arn,
               },
               Action: "sts:AssumeRole",
             },
           ],
         }),
         RoleName: roleName,
       }),
     ),
 );

 if (!Role) {
   throw new Error("Role not created");
 }

 // Create a policy that allows the user to list S3 buckets.
 const { Policy: listBucketPolicy } = await iamClient.send(
   new CreatePolicyCommand({
     PolicyDocument: JSON.stringify({
       Version: "2012-10-17",
       Statement: [
         {
           Effect: "Allow",
           Action: ["s3:ListAllMyBuckets"],
           Resource: "*",
         },
```

```
        ],
      }),
      PolicyName: policyName,
    }),
  );

  if (!listBucketPolicy) {
    throw new Error("Policy not created");
  }

  // Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
  await iamClient.send(
    new AttachRolePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
      RoleName: Role.RoleName,
    }),
  );

  // Assume the role.
  const stsClient = new STSClient({
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the assume role operation until it succeeds.
  const { Credentials } = await retry(
    { intervalInMs: 2000, maxRetries: 60 },
    () =>
      stsClient.send(
        new AssumeRoleCommand({
          RoleArn: Role.Arn,
          RoleSessionName: `iamBasicScenarioSession-${Math.floor(
            Math.random() * 1000000,
          )}`,
          DurationSeconds: 900,
        }),
      ),
  );

  if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
    throw new Error("Credentials not created");
  }
```

```
s3Client = new S3Client({
  credentials: {
    accessKeyId: Credentials.AccessKeyId,
    secretAccessKey: Credentials.SecretAccessKey,
    sessionToken: Credentials.SessionToken,
  },
});

// List the S3 buckets again.
// Retry the list buckets operation until it succeeds. AccessDenied might
// be thrown while the role policy is still stabilizing.
await retry({ intervalInMs: 2000, maxRetries: 120 }, () =>
  listBuckets(s3Client),
);

// Clean up.
await iamClient.send(
  new DetachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
);

await iamClient.send(
  new DeletePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
  }),
);

await iamClient.send(
  new DeleteRoleCommand({
    RoleName: Role.RoleName,
  }),
);

await iamClient.send(
  new DeleteAccessKeyCommand({
    UserName: userName,
    AccessKeyId,
  }),
);

await iamClient.send(
```

```
    new DeleteUserCommand({
      UserName: userName,
    }),
  );
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
    throw new Error("Buckets not listed");
  }

  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour JavaScript .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de l'utilisateur IAM.

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <username> <policyName> <roleName> <roleSessionName> <fileLocation>
 <bucketName>

    Where:
        username - The name of the IAM user to create.
        policyName - The name of the policy to create.
        roleName - The name of the role to create.
        roleSessionName - The name of the session required for the assumeRole
 operation.
        fileLocation - The file location to the JSON required to create the role
 (see Readme).
        bucketName - The name of the Amazon S3 bucket from which objects are
 read.
    """

    if (args.size != 6) {
        println(usage)
        exitProcess(1)
    }

    val userName = args[0]
    val policyName = args[1]
    val roleName = args[2]
    val roleSessionName = args[3]
    val fileLocation = args[4]
    val bucketName = args[5]

    createUser(userName)
```

```
    println("$userName was successfully created.")

    val polArn = createPolicy(policyName)
    println("The policy $polArn was successfully created.")

    val roleArn = createRole(roleName, fileLocation)
    println("$roleArn was successfully created.")
    attachRolePolicy(roleName, polArn)

    println("*** Wait for 1 MIN so the resource is available.")
    delay(60000)
    assumeGivenRole(roleArn, roleSessionName, bucketName)

    println("*** Getting ready to delete the AWS resources.")
    deleteRole(roleName, polArn)
    deleteUser(userName)
    println("This IAM Scenario has successfully completed.")
}

suspend fun createUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {
    val policyDocumentValue = """
    {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "s3:*"
                ],
                "Resource": "*"
            }
        ]
```

```
    }
    """.trimIndent()

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentValue
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}

suspend fun createRole(
    rolenameVal: String?,
    fileLocation: String?,
): String? {
    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request =
        CreateRoleRequest {
            roleName = rolenameVal
            assumeRolePolicyDocument = jsonObject.toJSONString()
            description = "Created using the AWS SDK for Kotlin"
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
```

```
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkMyList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String,
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(
    roleArnVal: String?,
    roleSessionNameVal: String?,
    bucketName: String,
) {
    val stsClient = StsClient.fromEnvironment { region = "us-east-1" }
```

```
    val roleRequest =
        AssumeRoleRequest {
            roleArn = roleArnVal
            roleSessionName = roleSessionNameVal
        }

    val roleResponse = stsClient.assumeRole(roleRequest)
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken

    val staticCredentials = StaticCredentialsProvider {
        accessKeyId = key
        secretAccessKey = secKey
        sessionToken = secToken
    }

    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 = S3Client.fromEnvironment {
        region = "us-east-1"
        credentialsProvider = staticCredentials
    }

    println("Created a S3Client using temp credentials.")
    println("Listing objects in $bucketName")

    val listObjects =
        ListObjectsRequest {
            bucket = bucketName
        }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
    }
}

suspend fun deleteRole(
    roleNameVal: String,
    polArn: String,
) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }
```

```
    // First the policy needs to be detached.
    val rolePolicyRequest =
        DetachRolePolicyRequest {
            policyArn = polArn
            roleName = roleNameVal
        }

    iam.detachRolePolicy(rolePolicyRequest)

    // Delete the policy.
    val request =
        DeletePolicyRequest {
            policyArn = polArn
        }

    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")

    // Delete the role.
    val roleRequest =
        DeleteRoleRequest {
            roleName = roleNameVal
        }

    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}

@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
```

```
    return jsonParser.parse(reader)
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour Kotlin.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
namespace Iam\Basics;

require 'vendor/autoload.php';

use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
```

```
use Aws\Sts\StsClient;
use Iam\IAMService;

echo("\n");
echo("---------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("---------------------------------------\n");

$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";

$key = $service->createAccessKey($user['UserName']);
$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
```

```
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
    ]);
    echo "this should not run";
} catch (S3Exception $exception) {
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
    echo "this should now not fail\n";
}

$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
```

```
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour PHP .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créer un utilisateur IAM et un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3. L'utilisateur n'a que le droit d'assumer le rôle. Après avoir assumé le rôle, utilisez des informations d'identification temporaires pour répertorier les compartiments pour le compte.

```
import json
import sys
import time
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError


def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
        time.sleep(1)
        print(".", end="")
        sys.stdout.flush()
    print()


def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, roles, and
 policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise
```

```
    try:
        user_key = user.create_access_key_pair()
        print(f"Created access key pair for user.")
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    try:
        role = iam_resource.create_role(
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
                    ],
                }
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        policy = iam_resource.create_policy(
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
```

```
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "s3:ListAllMyBuckets",
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
    except ClientError as error:
        print(
            f"Couldn't create a policy and attach it to role {role.name}. Here's
why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user.create_policy(
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
            f"Created an inline policy for {user.name} that lets the user assume
"
            f"the role."
        )
    except ClientError as error:
        print(
```

```
            f"Couldn't create an inline policy for user {user.name}. Here's why:
 "
            f"{error.response['Error']['Message']}"
        )
        raise

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, role


def show_access_denied_without_role(user_key):
    """
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
        "s3", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
        raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Attempt to list buckets with no permissions: AccessDenied.")
        else:
            raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the
 account.
    Uses the temporary credentials from the role to list the buckets that are
 owned
    by the assumed role's account.
```

```
    :param user_key: The access key of a user that has permission to assume the
role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id,
aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    # Create an S3 resource that can access the account with the temporary
credentials.
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
            f"Couldn't list buckets for the account. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise
```

```
def teardown(user, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        for user_pol in user.policies.all():
            user_pol.delete()
            print("Deleted inline user policy.")
        for key in user.access_keys.all():
            key.delete()
            print("Deleted user's access key.")
        user.delete()
        print(f"Deleted {user.name}.")
    except ClientError as error:
        print(
            "Couldn't delete user policy or delete user. Here's why: "
            f"{error.response['Error']['Message']}"
        )


def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
```

```
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
    role = None
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
        if user is not None and role is not None:
            teardown(user, role)
        print("Thanks for watching!")


if __name__ == "__main__":
    usage_demo()
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK for Python (Boto3).
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créer un utilisateur IAM et un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3. L'utilisateur n'a que le droit d'assumer le rôle. Après avoir assumé le rôle, utilisez des informations d'identification temporaires pour répertorier les compartiments pour le compte.

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client

  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Waits for the specified number of seconds.
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts('Give AWS time to propagate resources...')
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info('Tried and failed to create demo user.')
```

```
    @logger.info("\t#{e.code}: #{e.message}")
    @logger.info("\nCan't continue the demo without a user!")
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
  # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
  # @return [Aws::IAM::Role] The newly created role.
  def create_role(role_name, user)
    trust_policy = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Principal: { 'AWS': user.arn },
        Action: 'sts:AssumeRole'
      }]
    }.to_json
    role = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: trust_policy
    ).role
    @logger.info("Created role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
```

```
    @logger.info("Couldn't create a role for the demo. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end

  # Creates a policy that grants permission to list S3 buckets in the account,
and
  # then attaches the policy to a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param role [Aws::IAM::Role] The role that the policy is attached to.
  # @return [Aws::IAM::Policy] The newly created policy.
  def create_and_attach_role_policy(policy_name, role)
    policy_document = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Action: 's3:ListAllMyBuckets',
        Resource: 'arn:aws:s3:::*'
      }]
    }.to_json
    policy = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document
    ).policy
    @iam_client.attach_role_policy(
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an inline policy for a user that lets the user assume a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param user [Aws::IAM::User] The user that owns the policy.
```

```
 # @param role [Aws::IAM::Role] The role that can be assumed.
 # @return [Aws::IAM::UserPolicy] The newly created policy.
 def create_user_policy(policy_name, user, role)
   policy_document = {
     Version: '2012-10-17',
     Statement: [{
       Effect: 'Allow',
       Action: 'sts:AssumeRole',
       Resource: role.arn
     }]
   }.to_json
   @iam_client.put_user_policy(
     user_name: user.user_name,
     policy_name: policy_name,
     policy_document: policy_document
   )
   puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
 rescue Aws::Errors::ServiceError => e
   @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
   @logger.info("\t#{e.code}: #{e.message}")
   raise
 end

 # Creates an Amazon S3 resource with specified credentials. This is separated
into a
 # factory function so that it can be mocked for unit testing.
 #
 # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
 def create_s3_resource(credentials)
   Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
 end

 # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
 # Because the resource uses credentials with limited access, it may not be able
to
 # list the S3 buckets.
 #
 # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
 def list_buckets(s3_resource)
   count = 10
   s3_resource.buckets.each do |bucket|
```

```
      @logger.info "\t#{bucket.name}"
      count -= 1
      break if count.zero?
    end
  rescue Aws::Errors::ServiceError => e
    if e.code == 'AccessDenied'
      puts('Attempt to list buckets with no permissions: AccessDenied.')
    else
      @logger.info("Couldn't list buckets for the account. Here's why: ")
      @logger.info("\t#{e.code}: #{e.message}")
      raise
    end
  end

  # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
  # This is separated into a factory function so that it can be mocked for unit
testing.
  #
  # @param key_id [String] The ID of the access key used by the STS client.
  # @param key_secret [String] The secret part of the access key used by the STS
client.
  def create_sts_client(key_id, key_secret)
    Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
  end

  # Gets temporary credentials that can be used to assume a role.
  #
  # @param role_arn [String] The ARN of the role that is assumed when these
credentials
  #                          are used.
  # @param sts_client [AWS::STS::Client] An AWS STS client.
  # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
  def assume_role(role_arn, sts_client)
    credentials = Aws::AssumeRoleCredentials.new(
      client: sts_client,
      role_arn: role_arn,
      role_session_name: 'create-use-assume-role-scenario'
    )
    @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
    credentials
  end
```

```
  # Deletes a role. If the role has policies attached, they are detached and
  # deleted before the role is deleted.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    @iam_client.list_attached_role_policies(role_name:
 role_name).attached_policies.each do |policy|
      @iam_client.detach_role_policy(role_name: role_name, policy_arn:
 policy.policy_arn)
      @iam_client.delete_policy(policy_arn: policy.policy_arn)
      @logger.info("Detached and deleted policy #{policy.policy_name}.")
    end
    @iam_client.delete_role({ role_name: role_name })
    @logger.info("Role deleted: #{role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
 why:")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Deletes a user. If the user has inline policies or access keys, they are
 deleted
  # before the user is deleted.
  #
  # @param user [Aws::IAM::User] The user to delete.
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
```

```
  puts('-' * 88)
  puts('Welcome to the IAM create a user and assume a role demo!')
  puts('-' * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts('Try to list buckets with credentials for a user who has no permissions.')
  puts('Expect AccessDenied from this call.')
  scenario.list_buckets(
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key))
  )
  puts('Now, assume the role that grants permission.')
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key)
  )
  puts('Here are your buckets:')
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
  scenario.delete_user(user.user_name)
  puts('Thanks for watching!')
  puts('-' * 88)
rescue Aws::Errors::ServiceError => e
  puts('Something went wrong with the demo.')
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour Ruby .
  - AttachRolePolicy

- CreateAccessKey
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
        initialize_variables().await;
```

```
    if let Err(e) = run_iam_operations(
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (iamClient, String, String, String) {
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();

    (
        client,
        uuid,
```

```
        list_all_buckets_policy_document,
        inline_policy_document,
    )
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
    let key = iam_service::create_access_key(&client, user.user_name()).await?;

    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
    println!(
        "Created policy: {}",
        list_all_buckets_policy.policy_name.as_ref().unwrap()
```

```
    );

    let attach_role_policy_result =
        iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
            .await?;
    println!(
        "Attached the policy to the role: {:?}",
        attach_role_policy_result
    );

    let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
    let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
    iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
        .await?;
    println!("Created inline policy.");

    //First, fail to list the buckets with the user.
    let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
    let fail_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    println!("Fail config: {:?}", fail_config);
    let fail_client: s3Client = s3Client::new(&fail_config);
    match fail_client.list_buckets().send().await {
        Ok(e) => {
            println!("This should not run. {:?}", e);
        }
        Err(e) => {
            println!("Successfully failed with error: {:?}", e)
        }
    }

    let sts_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    let sts_client: stsClient = stsClient::new(&sts_config);
    sleep(Duration::from_secs(10)).await;
    let assumed_role = sts_client
```

```
        .assume_role()
        .role_arn(assume_role_role.arn())
        .role_session_name(format!("iam_demo_assumerole_session_{uuid}"))
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .access_key_id(),
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
                .as_ref()
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
```

```
            println!("This should now run successfully.")
        }
        Err(e) => {
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
    .await?;
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour Rust.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole

- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2

Les applications qui s'exécutent sur une instance Amazon EC2 doivent inclure des AWS informations d'identification dans les demandes d' AWS API. Vous pouvez demander à vos développeurs de stocker les AWS informations d'identification directement dans l'instance Amazon EC2 et d'autoriser les applications de cette instance à utiliser ces informations d'identification. Toutefois, dans ce cas, ils doivent gérer les informations d'identification, les transmettre en toute sécurité à chaque instance, puis mettre à jour chaque instance Amazon EC2 lorsqu'il convient d'effectuer une mise à jour des informations d'identification. Ceci représente beaucoup de travail supplémentaire.

Au lieu de cela, il est recommandé d'utiliser un rôle IAM pour gérer les informations d'identification temporaires pour les applications qui s'exécutent sur une instance Amazon EC2. Lorsque vous utilisez un rôle, vous n'avez pas besoin de distribuer d'informations d'identification à long terme (telles que des informations de connexion ou des clés d'accès) à une instance Amazon EC2. Au lieu de cela, le rôle fournit des autorisations temporaires que les applications peuvent utiliser lorsqu'elles appellent d'autres AWS ressources. Lorsque vous lancez une instance Amazon EC2, vous spécifiez un rôle IAM à associer à celle-ci. Les applications qui s'exécutent sur l'instance peuvent ensuite utiliser les informations d'identification de sécurité temporaires fournies pour le rôle pour signer les demandes d'API.

L'utilisation de rôles pour l'octroi d'autorisations à des applications qui s'exécutent sur des instances Amazon EC2 requiert une configuration supplémentaire. Une application exécutée sur une instance Amazon EC2 est extraite AWS par le système d'exploitation virtualisé. En raison de cette séparation supplémentaire, vous avez besoin d'une étape supplémentaire pour attribuer un AWS rôle et les autorisations associées à une instance Amazon EC2 et les mettre à la disposition de ses applications. Cette étape supplémentaire est la création d'un profil d'instance attaché à l'instance. Le profil d'instance contient le rôle et peut fournir les informations d'identification temporaires du rôle à une application qui s'exécute sur l'instance. Ces informations d'identification temporaires peuvent ensuite être utilisées dans les appels d'API de l'application pour accéder aux ressources et restreindre l'accès aux ressources spécifiées par le rôle uniquement.

> **Note**
>
> Chaque instance Amazon EC2 ne peut se voir attribuer qu'un seul rôle à la fois, et toutes les applications de l'instance partagent le même rôle et les mêmes autorisations. Lorsque vous exploitez Amazon ECS pour gérer vos instances Amazon EC2, vous pouvez attribuer des rôles aux tâches Amazon ECS qui peuvent être différenciés du rôle de l'instance Amazon EC2 sur laquelle elles s'exécutent. L'attribution d'un rôle à chaque tâche est conforme au principe de l'accès au moindre privilège et permet un contrôle plus précis des actions et des ressources.
> Pour plus d'informations, veuillez consulter la rubrique Using IAM roles with Amazon ECS tasks dans le Guide des bonnes pratiques pour Amazon Elastic Container Service.

Une telle utilisation des rôles présente plusieurs avantages. Dans la mesure où les informations d'identification des rôles sont temporaires et font automatiquement l'objet d'une mise à jour, vous n'avez pas à les gérer ni à vous soucier de risques de sécurité à long terme. Par ailleurs, si vous utilisez le même rôle pour plusieurs instances, toute modification apportée au rôle se propage automatiquement à toutes les instances.

> **Note**
>
> Même si un rôle est généralement attribué à une instance Amazon EC2 lors du lancement, un rôle peut également être attaché à une instance Amazon EC2 en cours d'exécution. Pour savoir comment attacher un rôle à une instance en cours d'exécution, consultez Rôles IAM pour Amazon EC2.

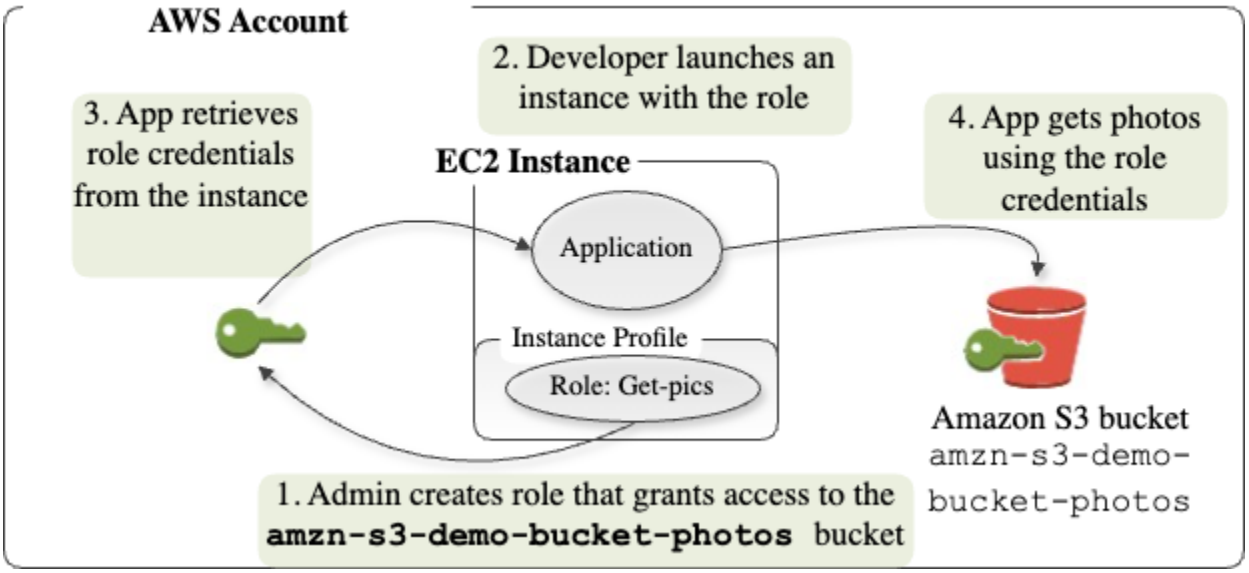
Rubriques

- Comment fonctionnent les rôles pour les instances Amazon EC2 ?
- Autorisations requises pour l'utilisation de rôles avec Amazon EC2
- Comment puis-je commencer ?
- Informations connexes

Comment fonctionnent les rôles pour les instances Amazon EC2 ?

Dans l'illustration suivante, un développeur exécute une application sur une instance Amazon EC2 qui requiert l'accès à un compartiment S3 nommé `amzn-s3-demo-bucket-photos`. Un

administrateur crée la fonction du service `Get-pics` et attache le rôle à l'instance Amazon EC2. Le rôle inclut une politique d'autorisations qui accorde un accès en lecture seule au compartiment S3 spécifié. Il inclut également une politique d'approbation qui permet à l'instance Amazon EC2 d'endosser le rôle et de récupérer les informations d'identification temporaires. Lorsque l'application s'exécute sur l'instance, elle peut accéder au compartiment photos en utilisant les informations d'identification temporaires du rôle. L'administrateur n'a pas besoin d'accorder au développeur l'autorisation d'accéder au compartiment photos et le développeur n'a à aucun moment besoin de partager ou gérer les informations d'identification.



1. L'administrateur utilise IAM pour créer le rôle **`Get-pics`**. Dans la politique d'approbation du rôle, l'administrateur spécifie que seules les instances Amazon EC2 peuvent endosser le rôle. Dans la politique d'autorisation du rôle, l'administrateur définit des autorisations en lecture seule pour le compartiment `amzn-s3-demo-bucket-photos`.
2. Un développeur lance une instance Amazon EC2 et lui affecte le rôle `Get-pics`.

> **Note**
>
> Si vous utilisez la console IAM, le profil d'instance est géré de manière quasiment transparente pour vous. Toutefois, si vous utilisez l'API AWS CLI or pour créer et gérer le rôle et l'instance Amazon EC2, vous devez créer le profil d'instance et lui attribuer le rôle au cours d'étapes distinctes. Ensuite, lorsque vous lancez l'instance, vous devez spécifier le nom du profil d'instance à la place du nom de rôle.

3. Lorsque l'application est en cours d'exécution, elle obtient des informations d'identification de sécurité temporaires à partir des métadonnées d'instance Amazon EC2, comme décrit dans Extraction des informations d'identification de sécurité à partir des métadonnées d'instance. Il s'agit d'informations d'identification de sécurité temporaires qui représentent le rôle et sont valides pendant une période limitée.

   Dans certains kits SDK AWS, le développeur peut utiliser un fournisseur qui gère de manière transparente les informations d'identification de sécurité temporaires. (La documentation de chaque AWS SDK décrit les fonctionnalités prises en charge par ce SDK pour la gestion des informations d'identification.)

   Sinon, l'application peut obtenir les informations d'identification temporaires directement à partir des métadonnées d'instance de l'instance Amazon EC2. Les informations d'identification et les valeurs associées se trouvent dans la catégorie `iam/security-credentials/`*`role-name`* (dans cet exemple, `iam/security-credentials/Get-pics`) des métadonnées. Si l'application extrait les informations d'identification des métadonnées de l'instance, elle peut les mettre en cache.
4. À l'aide des informations d'identification temporaires obtenues, l'application peut accéder au compartiment photos. Compte tenu de la politique attachée au rôle **`Get-pics`**, l'application dispose d'autorisations en lecture seule.

   Les informations d'identification de sécurité temporaires disponibles sur l'instance font automatiquement l'objet d'une mise à jour avant leur expiration, de manière à ce qu'un jeu d'identifiants valide soit toujours disponible. L'application doit simplement obtenir un nouvel ensemble d'informations d'identification à partir des métadonnées de l'instance avant l'expiration des informations d'identification actuelles. Il est possible d'utiliser le AWS SDK pour gérer les informations d'identification afin que l'application n'ait pas besoin d'inclure de logique supplémentaire pour actualiser les informations d'identification. Par exemple, instanciation de clients avec les fournisseurs d'informations d'identification de profil d'instance. En revanche, si l'application extrait les informations d'identification de sécurité temporaires des métadonnées d'instance, puis les met en cache, elle doit obtenir un ensemble d'informations d'identification actualisé toutes les heures, ou au moins 15 minutes avant l'expiration des informations d'identification actuelles. Le délai d'expiration est inclus dans les informations renvoyées dans la catégorie `iam/security-credentials/`*`role-name`*.

Autorisations requises pour l'utilisation de rôles avec Amazon EC2

Pour lancer une instance avec un rôle, le développeur doit avoir l'autorisation de lancer des instances Amazon EC2 et de transmettre des rôles IAM.

L'exemple de politique suivant permet aux utilisateurs d'utiliser le Console de gestion AWS pour lancer une instance avec un rôle. La politique inclut des caractères génériques (*) pour autoriser un utilisateur à transmettre n'importe quel rôle et à réaliser les actions Amazon EC2 énumérées. L'action `ListInstanceProfiles` permet aux utilisateurs d'afficher tous les rôles disponibles dans l' Compte AWS.

Example Exemple de politique qui autorise les utilisateurs à lancer une instance avec n'importe quel rôle à l'aide de la console Amazon EC2

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "IamPassRole",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        },
        {
            "Sid": "ListEc2AndListInstanceProfiles",
            "Effect": "Allow",
            "Action": [
                "iam:ListInstanceProfiles",
                "ec2:Describe*",
                "ec2:Search*",
                "ec2:Get*"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

Restreindre les rôles qui peuvent être transmis aux instances Amazon EC2 (en utilisant PassRole)

Vous pouvez utiliser l'autorisation `PassRole` pour spécifier les rôles qu'un utilisateur peut transmettre à une instance Amazon EC2 lors de son lancement. Cela évite que l'utilisateur exécute des applications ayant davantage d'autorisations qu'il n'en possède, autrement dit qu'il soit en mesure d'obtenir des droits élevés. Par exemple, supposons qu'une utilisatrice nommée Alice soit uniquement autorisée à lancer des instances Amazon EC2 et à utiliser des compartiments Amazon S3, mais que le rôle qu'elle transmet à une instance Amazon EC2 dispose d'autorisations permettant d'utiliser IAM et Amazon DynamoDB. Dans ce cas, il est possible qu'Alice soit en mesure de lancer l'instance, de s'y connecter, d'obtenir des informations d'identification de sécurité temporaires, puis d'effectuer plus d'actions IAM ou DynamoDB qu'elle n'est autorisée à le faire.

Pour limiter les rôles qu'un utilisateur peut transmettre à une instance Amazon EC2, vous pouvez créer une politique qui autorise l'action `PassRole`. Ensuite, vous attachez la politique à l'utilisateur (ou à un groupe IAM auquel il appartient) qui lancera les instances Amazon EC2. Dans l'élément `Resource` de la politique, vous répertoriez les rôles que l'utilisateur peut transmettre aux instances Amazon EC2. Lorsque l'utilisateur lance une instance et lui associe un rôle, Amazon EC2 vérifie si l'utilisateur est autorisé à transmettre ce rôle. Il est entendu que vous devez également vous assurer que le rôle que l'utilisateur est autorisé à transmettre n'inclut pas plus d'autorisations qu'il n'est censé en avoir.

> **Note**
>
> L'action d'API `PassRole` est différente de `RunInstances` ou `ListInstanceProfiles`. Il s'agit plutôt d'une autorisation qui AWS vérifie chaque fois qu'un ARN de rôle est transmis en tant que paramètre à une API (ou que la console le fait au nom de l'utilisateur). Elle permet à un administrateur de contrôler les rôles susceptibles d'être transmis et par quels utilisateurs. Dans le cas présent, elle veille à ce que l'utilisateur soit autorisé à attacher un rôle spécifique à une instance Amazon EC2.

Example Exemple de politique qui autorise un utilisateur à lancer une instance Amazon EC2 avec un rôle spécifique

Dans l'exemple suivant, la politique autorise les utilisateurs à lancer une instance avec un rôle à l'aide de l'API Amazon EC2. L'élément `Resource` spécifie l'Amazon Resource Name (ARN) d'un rôle. En spécifiant l'ARN, la politique accorde à l'utilisateur l'autorisation de transmettre uniquement le rôle `Get-pics`. Si l'utilisateur tente de spécifier un autre rôle lors du lancement d'une instance, l'action échoue. L'utilisateur est autorisé à exécuter n'importe quelle instance, qu'il s'agisse de transmettre un rôle ou non.

JSON

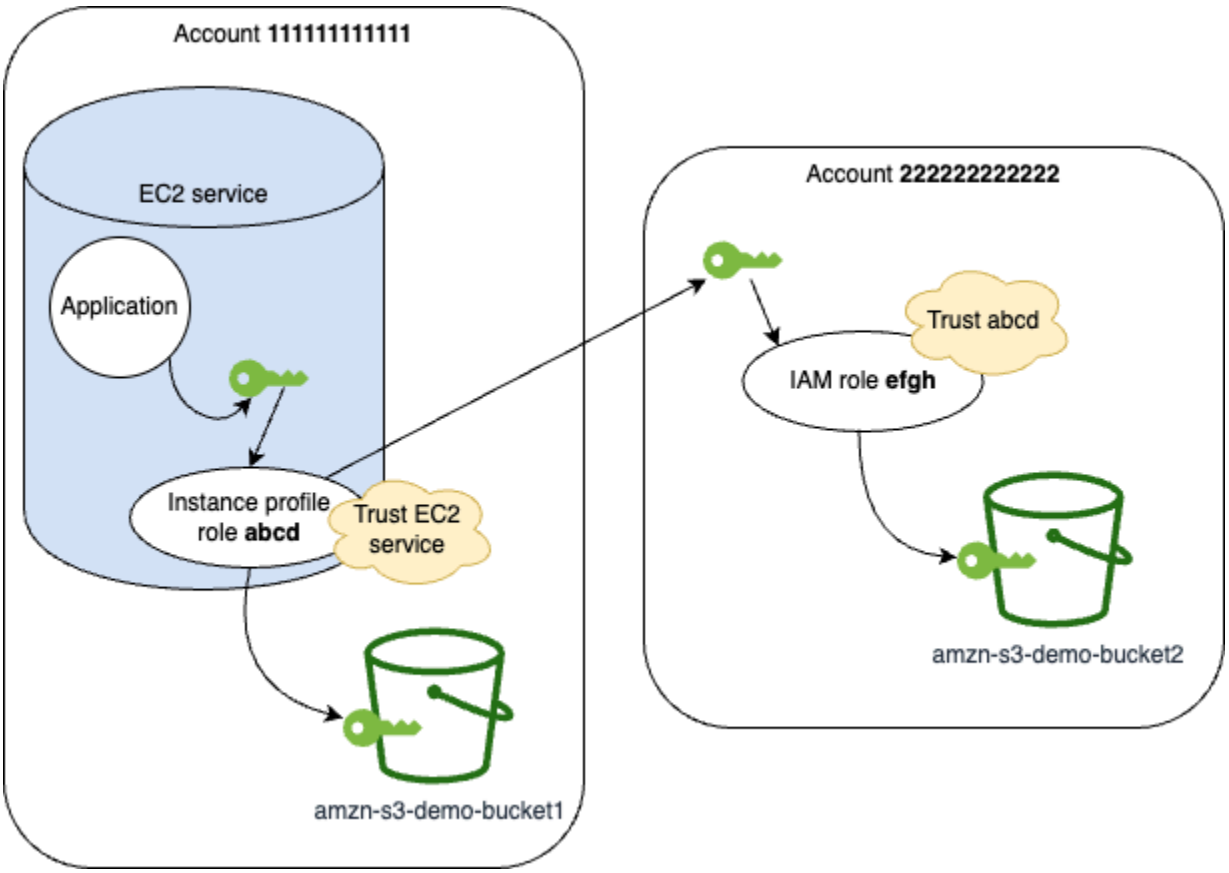
```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/Get-pics"
        }
    ]
}
```

Permettre à un rôle de profil d'instance de passer à un rôle dans un autre compte

Vous pouvez permettre à une application qui s'exécute sur une instance Amazon EC2 d'exécuter des commandes dans un autre compte. Pour ce faire, vous devez autoriser le rôle d'instance Amazon EC2 dans le premier compte pour passer à un rôle dans le second compte.

Imaginez que vous en utilisiez deux Comptes AWS et que vous souhaitiez autoriser une application exécutée sur une instance Amazon EC2 à exécuter des AWS CLI commandes sur les deux comptes. Supposons que l'instance Amazon EC2 existe dans le compte `111111111111`. Cette instance inclut le rôle de profil d'instance `abcd` qui permet à l'appli d'effectuer les tâches Amazon S3 `amzn-s3-demo-bucket1` en lecture seule sur le compartiment dans le même compte `111111111111`.

Toutefois, l'application doit également être autorisée à endosser le rôle `efgh` entre comptes pour accéder au compartiment `amzn-s3-demo-bucket2` Amazon S3 dans le compte `222222222222`.



Le rôle de profil d'instance Amazon EC2 `abcd` doit disposer des autorisations suivantes pour autoriser l'application à accéder au compartiment Amazon S3 `amzn-s3-demo-bucket1` :

Politique d'autorisations de rôle ***abcd*** du compte 111111111111

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
```

```
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1/*",
                "arn:aws:s3:::amzn-s3-demo-bucket1"
            ]
        },
        {
            "Sid": "AllowIPToAssumeCrossAccountRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

Le rôle `abcd` doit faire confiance au service Amazon EC2 pour endosser le rôle. Pour ce faire, le rôle `abcd` doit avoir la politique de confiance suivante :

Politique de confiance de rôle ***abcd*** du compte 111111111111

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "abcdTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"Service": "ec2.amazonaws.com"}
        }
    ]
```

```
}
```

Supposons que le rôle `efgh` entre comptes permet aux tâches Amazon S3 en lecture seule sur le compartiment `amzn-s3-demo-bucket2` dans le même compte `222222222222`. Pour ce faire, le rôle `efgh` entre comptes doit avoir la politique d'autorisations suivante :

Politique d'autorisations de rôle ***efgh*** du compte 222222222222

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket2/*",
                "arn:aws:s3:::amzn-s3-demo-bucket2"
            ]
        }
    ]
}
```

Le rôle `efgh` doit autoriser le rôle de profil d'instance `abcd` à l'endosser. Pour ce faire, le rôle `efgh` doit avoir la politique de confiance suivante :

Politique de confiance de rôle ***efgh*** du compte 222222222222

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

Comment puis-je commencer ?

Pour comprendre le fonctionnement des rôles avec les instances Amazon EC2, vous devez utiliser la console IAM pour créer un rôle, lancer une instance Amazon EC2 utilisant ce rôle, puis examiner l'instance en cours d'exécution. Vous pouvez examiner les métadonnées de l'instance pour voir comment les informations d'identification temporaires du rôle sont mises à la disposition d'une instance. Il est également possible d'étudier la façon dont une application exécutée sur l'instance peut utiliser un rôle donné. Consultez les ressources suivantes pour en savoir plus.

-
- Procédures SDK. La documentation du AWS SDK inclut des procédures détaillées qui montrent une application exécutée sur une instance Amazon EC2 qui utilise des informations d'identification temporaires pour que les rôles puissent lire un compartiment Amazon S3. Chacune des procédures suivantes comporte des étapes similaires, avec un langage de programmation différent :
  - Configuration de rôles IAM pour Amazon EC2 avec le kit SDK for Java dans le Manuel du développeur AWS SDK pour Java
  - Lancer une instance Amazon EC2 à l'aide du kit SDK pour .NET dans le Manuel du développeur AWS SDK pour .NET

- Création d'une instance Amazon EC2 avec le kit SDK for Ruby dans le Manuel du développeur AWS SDK pour Ruby

Informations connexes

Pour en savoir plus sur la création de rôles pour des instances Amazon EC2, consultez les informations suivantes :

- Pour de plus amples informations sur l'utilisation de rôles IAM avec des instances Amazon EC2, accédez au Guide de l'utilisateur Amazon EC2.
- Pour créer un rôle, consultez Création d'un rôle IAM
- Pour plus d'informations sur l'utilisation d'informations d'identification de sécurité temporaires, consultez Informations d'identification de sécurité temporaires dans IAM.
- Si vous utilisez l'API ou l'interface de ligne de commande (CLI) IAM vous devez créer et gérer les profils d'instance IAM. Pour plus d'informations sur les profils d'instance, consultez Utilisation des profils d'instance.
- Pour plus d'informations sur les informations d'identification de sécurité temporaires pour les rôles dans les métadonnées d'instance, consultez Extraction des informations d'identification de sécurité à partir des métadonnées d'instance dans le Guide de l'utilisateur Amazon EC2.

Utilisation des profils d'instance

Utilisez un profil d'instance pour transmettre un rôle IAM à une instance EC2. Pour de plus amples informations sur les rôles IAM, consultez Rôles IAM pour Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.

Gestion de profils d'instance (console)

Si vous utilisez le Console de gestion AWS pour créer un rôle pour Amazon EC2, la console crée automatiquement un profil d'instance et lui donne le même nom que le rôle. Lorsque vous utilisez ensuite la console Amazon EC2 pour lancer une instance avec un rôle IAM, vous pouvez sélectionner un rôle à associer à l'instance. Sur la console, la liste qui s'affiche est une liste de noms de profils d'instance. La console ne crée pas de profil d'instance pour un rôle qui n'est pas associé à Amazon EC2.

Vous pouvez utiliser le Console de gestion AWS pour supprimer des rôles IAM et des profils d'instance pour Amazon EC2 si le rôle et le profil d'instance portent le même nom. Pour en savoir

plus sur la suppression de profils d'instance, veuillez consulter Suppression de rôles ou de profils d'instance.

> **Note**
>
> Pour mettre à jour les autorisations d'une instance, remplacez son profil d'instance. Nous vous déconseillons de supprimer un rôle d'un profil d'instance, car cette modification peut prendre jusqu'à une heure avant de prendre effet.

Gestion des profils d'instance (AWS CLI or AWS API)

Si vous gérez vos rôles à partir de l'API AWS CLI ou de l' AWS API, vous créez des rôles et des profils d'instance en tant qu'actions distinctes. Étant donné que les rôles et les profils d'instance peuvent porter des noms différents, vous devez connaître les noms de vos profils d'instance, ainsi que ceux des rôles qu'ils contiennent. De cette façon, vous pouvez choisir le profil d'instance correct lorsque vous lancez une instance EC2.

Vous pouvez attacher des balises à vos ressources IAM, y compris les profils d'instance, pour les identifier, les organiser et contrôler l'accès. Vous pouvez baliser les profils d'instance uniquement lorsque vous utilisez l' AWS API AWS CLI or.

> **Note**
>
> Un profil d'instance peut contenir un seul rôle IAM, mais un rôle peut être inclus dans plusieurs profils d'instance. Cette limite est d'un rôle par profil d'instance ne peut pas être augmentée. Vous pouvez supprimer le rôle existant, puis ajouter un autre rôle à un profil d'instance. Vous devez ensuite attendre que la modification apparaisse dans l'ensemble pour des AWS raisons de cohérence éventuelle. Pour forcer la modification, vous devez dissocier le profil d'instance, puis associer le profil d'instance, ou vous pouvez arrêter votre instance, puis la redémarrer.

Gestion des profils d'instance (AWS CLI)

Vous pouvez utiliser les AWS CLI commandes suivantes pour utiliser les profils d'instance d'un AWS compte.

- Création d'un profil d'instance : `aws iam create-instance-profile`

- Balisage d'un profil d'instance : `aws iam tag-instance-profile`
- Affichage de la liste des balises d'un profil d'instance : `aws iam list-instance-profile-tags`
- Suppression des balises d'un profil d'instance : `aws iam untag-instance-profile`
- Ajout d'un rôle à un profil d'instance : `aws iam add-role-to-instance-profile`
- Affichage d'une liste de profils d'instance : `aws iam list-instance-profiles`, `aws iam list-instance-profiles-for-role`
- Obtention d'informations sur un profil d'instance : `aws iam get-instance-profile`
- Suppression d'un rôle d'un profil d'instance : `aws iam remove-role-from-instance-profile`
- Suppression d'un profil d'instance : `aws iam delete-instance-profile`

Vous pouvez également attacher un rôle à une instance EC2 déjà en cours d'exécution à l'aide des commandes suivantes. Pour de plus amples informations, veuillez consulter la section Rôles IAM pour Amazon EC2.

- Attachement d'un profil d'instance avec un rôle à une instance EC2 arrêtée ou en cours d'exécution : `aws ec2 associate-iam-instance-profile`
- Obtention d'informations sur un profil d'instance attaché à une instance EC2 : `aws ec2 describe-iam-instance-profile-associations`
- Détachement d'un profil d'instance avec un rôle d'une instance EC2 arrêtée ou en cours d'exécution : `aws ec2 disassociate-iam-instance-profile`

Gestion des profils d'instance (AWS API)

Vous pouvez appeler les opérations d' AWS API suivantes pour travailler avec des profils d'instance dans un Compte AWS.

- Création d'un profil d'instance : `CreateInstanceProfile`
- Balisage d'un profil d'instance : `TagInstanceProfile`
- Affichage de la liste des balises d'un profil d'instance : `ListInstanceProfileTags`
- Suppression des balises d'un profil d'instance : `UntagInstanceProfile`
- Ajout d'un rôle à un profil d'instance : `AddRoleToInstanceProfile`

- Affichage d'une liste de profils d'instance : `ListInstanceProfiles`, `ListInstanceProfilesForRole`
- Obtention d'informations sur un profil d'instance : `GetInstanceProfile`
- Suppression d'un rôle d'un profil d'instance : `RemoveRoleFromInstanceProfile`
- Suppression d'un profil d'instance : `DeleteInstanceProfile`

Vous pouvez également attacher un rôle à une instance EC2 déjà en cours d'exécution en appelant les opérations suivantes. Pour de plus amples informations, veuillez consulter la section Rôles IAM pour Amazon EC2.

- Attachement d'un profil d'instance avec un rôle à une instance EC2 arrêtée ou en cours d'exécution : `AssociateIamInstanceProfile`
- Obtention d'informations sur un profil d'instance attaché à une instance EC2 : `DescribeIamInstanceProfileAssociations`
- Détachement d'un profil d'instance avec un rôle d'une instance EC2 arrêtée ou en cours d'exécution : `DisassociateIamInstanceProfile`

Fournisseurs d'identité et fédération au sein AWS

À titre de meilleure pratique, nous vous recommandons de demander aux utilisateurs humains d'utiliser la fédération avec un fournisseur d'identité pour accéder aux AWS ressources au lieu de créer des utilisateurs IAM individuels dans votre Compte AWS. Avec un fournisseur d'identité (IdP), vous pouvez gérer les identités de vos utilisateurs en dehors de AWS celui-ci et leur accorder les autorisations nécessaires pour utiliser les AWS ressources de votre compte. Ceci est utile si votre organisation dispose déjà de son propre système d'identité, par exemple un répertoire d'utilisateurs d'entreprise. Il est également utile si vous créez une application mobile ou une application Web nécessitant l'accès à AWS des ressources.

> **Note**
>
> Vous pouvez également gérer les utilisateurs humains dans l'IAM Identity Center avec un fournisseur d'identité SAML externe plutôt que d'utiliser la fédération SAML dans IAM. La fédération IAM Identity Center avec un fournisseur d'identité vous permet de donner aux utilisateurs l'accès à plusieurs AWS comptes de votre organisation et à de multiples AWS applications. Pour en savoir plus sur les situations spécifiques dans lesquelles un utilisateur IAM est nécessaire, veuillez consulter Quand créer un utilisateur IAM (au lieu d'un rôle).

Vous pouvez également utiliser le gestionnaire d'accès aux comptes, une fonctionnalité IAM qui vous permet d'attribuer des rôles IAM existants sur les comptes de votre organisation aux utilisateurs et aux groupes IAM Identity Center. Vous pouvez l'utiliser conjointement avec des ensembles d'autorisations ou seul.

Si vous préférez utiliser un seul AWS compte sans activer IAM Identity Center, vous pouvez utiliser IAM avec un IdP externe qui fournit des informations d'identité à l' AWS aide d'OpenID Connect (OIDC) ou de SAML 2.0 (Security Assertion Markup Language 2.0). OIDC connecte des applications, telles que GitHub des actions, qui ne s'exécutent pas AWS à AWS des ressources. Voici quelques exemples de fournisseurs d'identité SAML bien connus : Shibboleth et Active Directory Federation Services.

Lorsque vous utilisez un fournisseur d'identité , vous n'avez pas à créer de code de connexion personnalisé ni à gérer vos propres identités utilisateur. L'IdP prévoit cela pour vous. Vos utilisateurs externes se connectent via un IdP et vous pouvez autoriser ces identités externes à utiliser les AWS ressources de votre compte. Les fournisseurs d'identité contribuent à Compte AWS assurer votre sécurité, car vous n'avez pas à distribuer ou à intégrer des informations de sécurité à long terme, telles que des clés d'accès, dans votre application.

Consultez le tableau suivant pour déterminer le type de fédération IAM le mieux adapté à votre cas d'utilisation : IAM, IAM Identity Center ou Amazon Cognito. Les résumés et le tableau suivants fournissent une vue d'ensemble des méthodes que vos utilisateurs peuvent utiliser pour obtenir un accès fédéré aux AWS ressources.

Type de fédération IAM	Account type (Type de compte)	Gestion de l'accès de…	Source d'identité prise en charge
Fédération avec IAM Identiy Center	Comptes multiples gérés par AWS Organizations	Les utilisateurs humains de votre personnel	• SAML 2.0 • Managed Active Directory • Répertoire d'Identity Center
Fédération avec IAM	Compte unique et autonome	• Utilisateurs humains dans le cadre de déploieme	• SAML 2.0 • OIDC

Type de fédération IAM	Account type (Type de compte)	Gestion de l'accès de…	Source d'identité prise en charge
		nts à court terme et à petite échelle • Utilisateurs machines	
Fédération avec les réserves d'identités Amazon Cognito	N'importe lequel	Les utilisateurs d'applications qui nécessitent une autorisation IAM pour accéder aux ressources	• SAML 2.0 • OIDC • Sélectionner les fournisseurs d'identité sociaux OAuth 2.0

Fédération avec IAM Identiy Center

Pour une gestion centralisée des accès des utilisateurs humains, nous vous recommandons d'utiliser IAM Identity Center pour gérer l'accès à vos comptes et les autorisations au sein de ceux-ci. Les utilisateurs d'IAM Identity Center reçoivent des informations d'identification de courte durée pour accéder à vos AWS ressources. Vous pouvez utiliser Active Directory, un fournisseur d'identité externe (IdP) ou un annuaire IAM Identity Center comme source d'identité permettant aux utilisateurs et aux groupes d'attribuer l'accès à vos AWS ressources.

IAM Identity Center prend en charge la fédération d'identités avec SAML (Security Assertion Markup Language) 2.0 afin de fournir un accès d'authentification unique fédéré aux utilisateurs autorisés à utiliser les applications du portail d'accès. AWS Les utilisateurs peuvent ensuite accéder à des services prenant en charge le protocole SAML, notamment à Console de gestion AWS des applications tierces telles que Microsoft 365, SAP Concur et Salesforce.

Fédération avec IAM

Bien que nous recommandions fortement de gérer les utilisateurs humains dans IAM Identity Center, vous pouvez activer l'accès principal fédéré avec IAM pour les utilisateurs humains dans le cadre de déploiements à court terme et à petite échelle. IAM vous permet d'utiliser SAML 2.0 et Open ID Connect (OIDC) IdPs séparément et d'utiliser des attributs principaux fédérés pour le contrôle d'accès. Avec IAM, vous pouvez transmettre des attributs utilisateur, tels que le centre de coûts, le

titre ou les paramètres régionaux, de votre adresse IdPs à AWS et implémenter des autorisations d'accès affinées en fonction de ces attributs.

Une charge de travail est un ensemble de ressources et de code qui fournit une valeur business, par exemple une application destinée au client ou un processus de backend. Votre charge de travail peut nécessiter une identité IAM pour adresser des demandes à AWS des services, des applications, des outils opérationnels et des composants. Ces identités incluent les machines qui s'exécutent dans vos AWS environnements, telles que les instances ou les AWS Lambda fonctions Amazon EC2.

Vous pouvez également gérer les Identités machine pour les parties externes qui ont besoin d'un accès. Pour donner accès aux identités des machines, vous pouvez utiliser des rôles IAM. Les rôles IAM disposent d'autorisations spécifiques et fournissent un moyen d'accès AWS en s'appuyant sur des informations d'identification de sécurité temporaires associées à une session de rôle. En outre, il se peut AWS que des machines extérieures aient besoin d'accéder à vos AWS environnements. Pour les machines qui s'exécutent en dehors de AWS vous, vous pouvez utiliser IAM Roles Anywhere. Pour plus d'informations sur les rôles , consultez Rôles IAM. Pour plus de détails sur l'utilisation des rôles pour déléguer l'accès entre Comptes AWS eux, consultezTutoriel IAM : accès délégué via AWS comptes utilisant des rôles IAM.

Pour lier un IdP directement à IAM, vous créez une entité fournisseur d'identité afin d'établir une relation de confiance entre vous Compte AWS et l'IdP. Supports IAM IdPs compatibles avec OpenID Connect (OIDC) ou SAML 2.0 (Security Assertion Markup Language 2.0). Pour plus d'informations sur l'utilisation de l'un d' IdPs entre eux avec AWS, consultez les sections suivantes :

- Fédération OIDC
- Fédération SAML 2.0

Fédération avec les réserves d'identités Amazon Cognito

Amazon Cognito est conçu pour les développeurs qui souhaitent authentifier et autoriser les utilisateurs dans leurs applications mobiles et Web. Les groupes d'utilisateurs Amazon Cognito ajoutent des fonctionnalités de connexion et d'inscription à votre application, et les réserves d'identités fournissent des informations d'identification IAM qui permettent à vos utilisateurs d'accéder aux ressources protégées que vous gérez dans AWS. Les réserves d'identités obtiennent des informations d'identification pour les sessions temporaires par le biais de l'opération API `AssumeRoleWithWebIdentity`.

Amazon Cognito travaille avec des fournisseurs d'identité externes qui prennent en charge SAML et OpenID Connect, ainsi qu'avec des fournisseurs d'identité sociaux tels que Facebook, Google et Amazon. Votre application peut connecter un utilisateur appartenant à un groupe d'utilisateurs ou à un IdP externe, puis récupérer des ressources en son nom grâce à des sessions temporaires personnalisées dans un rôle IAM.

Ressources supplémentaires

- Pour une démonstration de la création d'un proxy de fédération personnalisé qui permet l'authentification unique (SSO) à l' Console de gestion AWS aide du système d'authentification de votre organisation, consultez. Activez l'accès personnalisé du courtier d'identité au AWS console
- Pour obtenir des instructions sur la manière d'autoriser vos AWS charges de travail à accéder à des services externes sans stocker d'informations d'identification, consultezFédérer AWS Identités vis-à-vis des services externes.

Scénarios courants

> **Note**
>
> Nous vous recommandons de demander à vos utilisateurs humains d'utiliser des informations d'identification temporaires lors de l'accès AWS. Avez-vous envisagé de l'utiliser AWS IAM Identity Center ? Vous pouvez utiliser IAM Identity Center pour gérer de manière centralisée l'accès à plusieurs comptes Comptes AWS et fournir aux MFA-protected utilisateurs un accès en authentification unique à tous les comptes qui leur sont attribués depuis un seul endroit. Avec IAM Identity Center, vous pouvez créer et gérer les identités des utilisateurs dans IAM Identity Center ou vous connecter facilement à votre fournisseur d'identité compatible SAML 2.0 existant. Pour plus d'informations, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Vous pouvez utiliser un fournisseur d'identité (IdP) externe pour gérer les identités des utilisateurs à l'extérieur d' AWS et de l'IdP externe. Un IdP externe peut fournir des informations d'identité à l' AWS aide d'OpenID Connect (OIDC) ou du langage SAML (Security Assertion Markup Language). L'OIDC est couramment utilisé lorsqu'une application qui ne s'exécute pas AWS a besoin d'accéder à AWS des ressources.

Lorsque vous souhaitez configurer la fédération avec un IdP externe, vous créez un fournisseur d'identité IAM pour fournir des AWS informations sur l'IdP externe et sa configuration. Cela permet d'établir la confiance entre votre IdP Compte AWS et l'IdP externe. Les rubriques suivantes présentent des scénarios courants d'utilisation des fournisseurs d'identité IAM.

Rubriques

- Amazon Cognito pour les applications mobiles
- Fédération OIDC pour applications mobiles

Amazon Cognito pour les applications mobiles

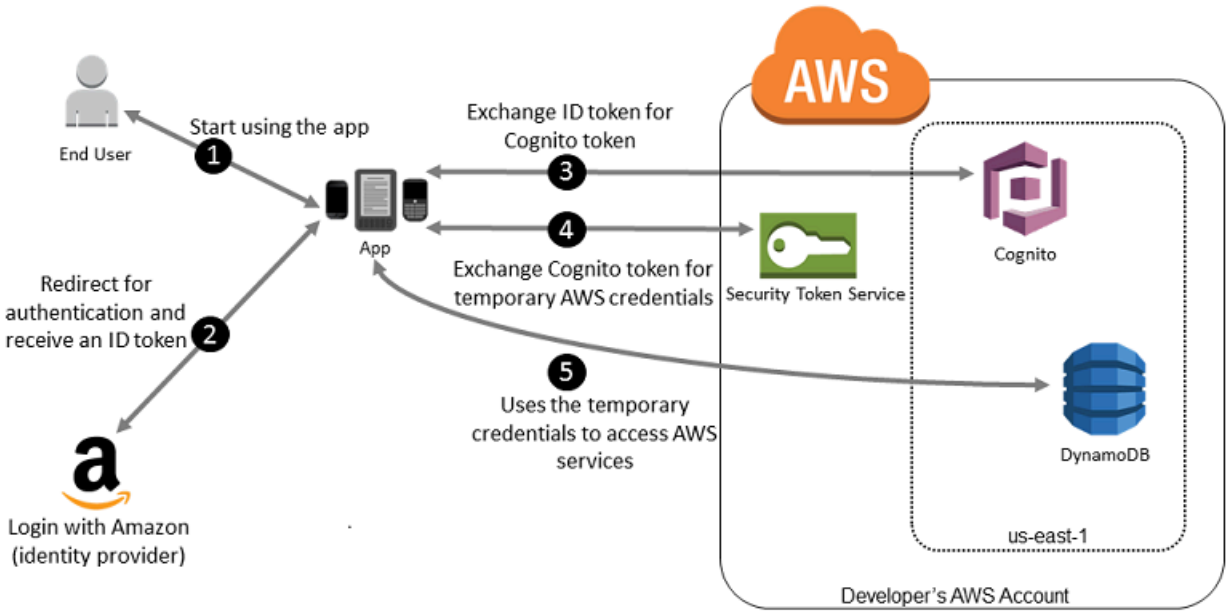
Le meilleur moyen d'utiliser la fédération OIDC consiste à utiliser Amazon Cognito. Par exemple, la développeuse Adèle est en train de créer un jeu pour appareil mobile dans lequel les données utilisateur, telles que les scores et les profils, sont stockées dans Amazon S3 et Amazon DynamoDB. Adèle pourrait également stocker ces données en local sur l'appareil et utiliser Amazon Cognito pour les maintenir synchronisées sur les différents appareils. Elle sait que pour des raisons de sécurité et de maintenance, des informations d'identification (credentials) de sécurité AWS à long terme ne doivent pas être distribuées avec le jeu. Elle sait également que le jeu pourrait avoir un grand nombre d'utilisateurs. Pour toutes ces raisons, elle ne veut pas créer de nouvelles identités dans IAM pour chaque joueur. Au lieu de cela, elle développe le jeu de sorte que les utilisateurs puissent se connecter à l'aide d'une identité qu'ils ont déjà établie avec un fournisseur d'identité (IdP) externe connu, comme Login with Amazon, Facebook, Google, ou tout fournisseur d'identité compatible OpenID Connect (OIDC). Son jeu peut tirer parti du mécanisme d'authentification de l'un de ces fournisseurs afin de valider l'identité de l'utilisateur.

Pour permettre à l'application mobile d'accéder à ses AWS ressources, Adele enregistre d'abord un identifiant de développeur avec l'identifiant de son choix IdPs. Elle configure également l'application avec chacun de ces fournisseurs. Dans Compte AWS cette section qui contient le compartiment Amazon S3 et la table DynamoDB du jeu, Adele utilise Amazon Cognito pour créer des rôles IAM qui définissent précisément les autorisations dont le jeu a besoin. Si elle utilise un IdP OIDC, elle crée également une entité fournisseur d'identité OIDC IAM pour établir la confiance entre un pool d'identités Amazon Cognito en elle et l'IdP. Compte AWS

Dans le code de l'application, Adèle appelle l'interface de la connexion pour le fournisseur d'identité l'IdP qu'elle a configuré précédemment. Le fournisseur d'identité traite tous les détails permettant à l'utilisateur de se connecter, et l'application obtient un jeton d'accès OAuth ou jeton d'ID OIDC du fournisseur. L'application d'Adele peut échanger ces informations d'authentification contre un

ensemble d'informations de sécurité temporaires comprenant un identifiant de clé d' AWS accès, une clé d'accès secrète et un jeton de session. L'application peut ensuite utiliser ces informations d'identification pour accéder aux services Web proposés par AWS. L'application est limitée aux autorisations qui sont définies dans le rôle qu'elle endosse.

La figure suivante illustre un flux simplifié d'un fonctionnement possible, à l'aide de Login with Amazon comme fournisseur d'identité. Pour l'étape 2, l'application peut également utiliser Facebook, Google ou tout autre OIDC-compatible IdP, mais cela n'est pas illustré ici.



1. Un client démarre votre application sur un appareil mobile. L'application demande à l'utilisateur de se connecter.
2. L'application utilise des ressources Login with Amazon pour accepter les informations d'identification de l'utilisateur.
3. L'application utilise des opérations d'API Amazon Cognito `GetId` et `GetCredentialsForIdentity` pour échanger le jeton d'ID Login with Amazon (Connexion avec Amazon) contre un jeton Amazon Cognito. Amazon Cognito, qui a été configuré pour approuver votre projet Login with Amazon (Connexion avec Amazon), génère un jeton qu'il échange contre des informations d'identification de session temporaires avec AWS STS.
4. L'application reçoit des informations d'identification de sécurité temporaires d'Amazon Cognito. Votre application peut également utiliser le flux de travail de base (classique) d'Amazon Cognito pour récupérer les jetons à AWS STS utiliser`AssumeRoleWithWebIdentity`. Pour plus

d'informations, consultez la rubrique Flux d'authentification des groupes d'identités (identités fédérées) dans le Guide du développeur Amazon Cognito.

5. Les informations d'identification de sécurité temporaires peuvent être utilisées par l'application pour accéder aux ressources AWS requises par l'application pour fonctionner. Le rôle associé aux informations d'identification de sécurité temporaires et les politiques qui lui sont attribuées détermine ce qui est accessible.

Suivez la procédure suivante pour configurer votre application afin qu'elle utilise Amazon Cognito pour authentifier les utilisateurs et permettre à votre application d'accéder aux AWS ressources. Pour les étapes spécifiques permettant de réaliser ce scénario, consultez la documentation d'Amazon Cognito.

1. (Facultatif) Connectez-vous en tant que développeur auprès de Login with Amazon, Facebook, Google ou tout autre fournisseur d'identité compatible OpenID Connect (OIDC) et configurez une ou plusieurs applications avec le fournisseur. Cette étape est facultative, car Amazon Cognito prend également en charge l'accès non authentifié (invité) pour vos utilisateurs.
2. Accédez à Amazon Cognito dans le Console de gestion AWS. Utilisez l'assistant Amazon Cognito pour créer un groupe d'identités, qui est un conteneur utilisé par Amazon Cognito pour maintenir des identités d'utilisateur final organisées pour vos applications. Vous pouvez partager des pools d'identité entre des applications. Lorsque vous configurez un groupe d'identités, Amazon Cognito crée un ou deux rôles IAM (un pour les identités authentifiées et un pour les identités « invitées » non authentifiées) qui définissent des autorisations pour les utilisateurs d'Amazon Cognito.
3. Intégrez AWSAmplify avec votre application et importez les fichiers requis pour utiliser Amazon Cognito.
4. Créez une instance du fournisseur d'informations d'identification Amazon Cognito en transmettant l'ID de groupe d'identité, votre numéro d' Compte AWS et l'Amazon Resource Name (ARN) des rôles que vous avez associés au groupe d'identités. L'assistant Amazon Cognito Console de gestion AWS fournit un exemple de code pour vous aider à démarrer.
5. Lorsque votre application accède à une AWS ressource, transmettez l'instance du fournisseur d'informations d'identification à l'objet client, qui transmet des informations d'identification de sécurité temporaires au client. Les autorisations pour les informations d'identification sont basées sur le ou les rôles que vous avez définis précédemment.

Pour plus d’informations, consultez les ressources suivantes :

- Connectez-vous (Android) dans la documentation du AWS Amplify Framework.
- Connectez-vous (iOS) dans la documentation du AWS Amplify Framework.

Fédération OIDC pour applications mobiles

Vous obtiendrez les meilleurs résultats en utilisant Amazon Cognito comme fournisseur d'identité dans la quasi-totalité des scénarios de fédération OIDC. Amazon Cognito est facile à utiliser et offre des capacités supplémentaires, comme l'accès anonyme (sans authentification) et la synchronisation des données utilisateur entre les périphériques et les fournisseurs. Cependant, si vous avez déjà créé une application exploitant la fédération OIDC en appelant manuellement l'API `AssumeRoleWithWebIdentity`, vous pouvez continuer à l'utiliser : vos applications fonctionneront malgré tout.

L'utilisation de la fédération OIDC sans Amazon Cognito se déroule comme suit :

1. Inscrivez-vous en tant que développeur auprès de l'IdP externe et configurez votre application avec l'ID unique qu'il vous donnera. (Différents IdPs utilisent une terminologie différente pour ce processus. Ce schéma utilise le terme configure pour désigner le processus d'identification de votre application auprès de l'IdP.) Chaque IdP vous donne un identifiant d'application qui lui est propre. Ainsi, si vous configurez la même application avec plusieurs IdPs, votre application aura plusieurs ID d'application. Vous pouvez configurer plusieurs applications auprès de chaque fournisseur.

   Les liens externes suivants fournissent des informations sur l'utilisation de certains fournisseurs d'identité couramment utilisés (IdPs) :

   - Login with Amazon Developer Center
   - Add Facebook Login to Your App or Website sur le site des développeurs Facebook.
   - Using OAuth 2.0 for Login (OpenID Connect) sur le site des développeurs Google.

   > **Important**
   >
   > Si vous utilisez un fournisseur d'identité OIDC de Google, Facebook ou Amazon Cognito, ne créez pas de fournisseur d'identité IAM distinct dans le. Console de gestion AWS AWS intègre ces fournisseurs d'identité OIDC et les met à votre disposition. Ignorez l'étape suivante et passez directement à la création de rôles à l'aide de votre fournisseur d'identité.

2. Si vous utilisez un IdP autre que Google, Facebook ou Amazon Cognito, compatible avec OIDC, créez une entité de fournisseur d'identité IAM pour lui.
3. Dans IAM, créez un ou plusieurs rôles. Pour chaque rôle, définissez qui peut endosser le rôle (politique d'approbation) et les autorisations accordées aux utilisateurs de l'application (politique d'autorisation). En général, vous créez un rôle pour chaque IdP pris en charge par l'application. Par exemple, vous pouvez créer un rôle qui est endossé par une application si l'utilisateur se connecte via Login with Amazon, un deuxième rôle pour la même application s'il se connecte via Facebook, et un troisième rôle s'il se connecte via Google. Pour la relation de confiance, spécifiez l'IdP (similaire Amazon.com) comme `Principal` (l'entité de confiance) et incluez un identifiant `Condition` qui correspond à l'ID d'application attribué à l'IdP. Des exemples de rôles pour différents fournisseurs sont présentés dans la rubrique Créer un rôle pour un fournisseur d'identité tiers.
4. Dans votre application, authentifiez vos utilisateurs en recourant à l'IdP. La procédure à suivre varie en fonction de l'IdP utilisé (Login with Amazon, Facebook ou Google) et la plateforme sur laquelle s'exécute votre application. Par exemple, la méthode d'authentification d'une application Android peut être différente de celle d'une application iOS ou d'une application JavaScript-based Web.

   En règle générale, si l'utilisateur n'est pas déjà connecté, l'IdP affiche une page de connexion. Une fois qu'il a authentifié l'utilisateur, l'IdP retourne un jeton d'authentification contenant des informations sur l'utilisateur à l'application. Ces informations dépendent de ce que l'IdP expose et des informations que l'utilisateur est disposé à partager. Vous pouvez utiliser ces informations dans votre application.
5. Dans l'application, effectuez un appel non signé à l'action `AssumeRoleWithWebIdentity` pour solliciter des informations d'identification de sécurité temporaires. Dans la demande, vous transmettez le jeton d'authentification de l'IdP et vous spécifiez l'Amazon Resource Name (ARN) pour le rôle IAM que vous avez créé pour cet IdP. AWS vérifie que le jeton est fiable et valide et, le cas échéant, renvoie les informations d'identification de sécurité temporaires à votre application qui disposent des autorisations pour le rôle que vous nommez dans la demande. La réponse inclut également des métadonnées se rapportant à l'utilisateur provenant de l'IdP telles que l'ID utilisateur unique que l'IdP associe à l'utilisateur.
6. À l'aide des informations d'identification de sécurité temporaires fournies dans la `AssumeRoleWithWebIdentity` réponse, votre application envoie des demandes signées aux opérations de AWS l'API. Les informations d'identification de l'utilisateur provenant de l'IdP permettent de distinguer les utilisateurs de votre application. Par exemple, vous pouvez placer des objets dans des dossiers Amazon S3 qui incluent l'identifiant de l'utilisateur comme préfixe

ou suffixe. Ceci vous permet de créer des politiques d'accès qui verrouillent un dossier, afin que seul l'utilisateur ayant l'ID approprié soit autorisé à y accéder. Pour de plus amples informations, veuillez consulter AWS STS principes d'utilisateur fédérés.

7. Votre application met généralement en cache ces informations d'identification de sécurité temporaires afin que vous n'ayez pas à en redemander à chaque fois que l'application doit envoyer une demande à AWS. Par défaut, les informations d'identification restent valides pendant une heure. Lorsque les informations d'identification expirent (ou avant le délai d'expiration), vous effectuez un autre appel à `AssumeRoleWithWebIdentity` pour obtenir un nouvel ensemble d'informations d'identification de sécurité temporaires. Selon l'IdP utilisé et la façon dont il gère ses jetons, il sera peut-être nécessaire d'actualiser le jeton de l'IdP avant un nouvel appel à `AssumeRoleWithWebIdentity`, car les jetons de l'IdP expirent généralement après un délai spécifié. Si vous utilisez le AWS SDK pour iOS ou le AWS SDK pour Android, vous pouvez utiliser l'AmazonSTSCredentialsProvideraction, qui gère les informations d'identification temporaires IAM, notamment en les actualisant si nécessaire.

Fédération OIDC

Imaginez que vous créez une application qui accède à AWS des ressources, comme des GitHub actions qui utilise des flux de travail pour accéder à Amazon S3 et DynamoDB.

Lorsque vous utilisez ces flux de travail, vous envoyez des demandes à AWS des services qui doivent être signés à l'aide d'une clé d' AWS accès. Cependant, nous vous recommandons vivement de ne pas stocker les AWS informations d'identification à long terme dans des applications extérieures AWS. Configurez plutôt vos applications pour demander des informations d'identification de AWS sécurité temporaires de manière dynamique lorsque cela est nécessaire à l'aide de la fédération OIDC. Les informations d'identification temporaires fournies correspondent à un AWS rôle qui ne dispose que des autorisations nécessaires pour effectuer les tâches requises par l'application.

Lors de l'utilisation de la fédération OIDC, il n'est pas nécessaire de créer de code de connexion personnalisé ni de gérer vos propres identités utilisateur. Vous pouvez plutôt utiliser OIDC dans des applications, telles que GitHub Actions ou tout autre IdP compatible avec OpenID Connect (OIDC), pour vous authentifier. AWS Ils reçoivent un jeton d'authentification, appelé jeton Web JSON (JWT), puis échangent ce jeton contre des informations d'identification de sécurité temporaires dans AWS cette carte vers un rôle IAM autorisé à utiliser des ressources spécifiques dans votre. Compte AWS L'utilisation d'un IdP vous aide à assurer votre Compte AWS sécurité, car vous n'avez pas à intégrer et à distribuer des informations de sécurité à long terme dans votre application.

La fédération OIDC prend en charge à la fois l'authentification de machine à machine (telle que les CI/CD pipelines, les scripts automatisés et les applications sans serveur) et l'authentification des utilisateurs humains. Pour les scénarios d'authentification des utilisateurs humains dans lesquels vous devez gérer l'inscription, la connexion et les profils des utilisateurs, envisagez d'utiliser Amazon Cognito comme fournisseur d'identité. Pour en savoir plus sur l'utilisation d'Amazon Cognito avec OIDC, consultez Amazon Cognito pour les applications mobiles.

> **Note**
>
> Les jetons Web JSON (JWT) émis par les fournisseurs d'identité OpenID Connect (OIDC) contiennent un délai d'expiration dans la demande `exp` qui indique la date d'expiration du jeton. IAM fournit une fenêtre de cinq minutes au-delà du délai d'expiration spécifié dans le JWT pour tenir compte du décalage d'horloge, comme le permet la norme OpenID Connect (OIDC) Core 1.0. Cela signifie que les JWT OIDC reçus par IAM après le délai d'expiration mais dans cette fenêtre de cinq minutes sont acceptés pour une évaluation et un traitement ultérieurs.

Rubriques

- Ressources supplémentaires pour la fédération OIDC
- Création d'un fournisseur d'identité OpenID Connect (OIDC) dans IAM
- Obtention de l'empreinte numérique d'un fournisseur d'identité OpenID Connect
- Identity-provider contrôles pour les fournisseurs OIDC partagés

Ressources supplémentaires pour la fédération OIDC

Les ressources suivantes peuvent vous aider à en savoir plus sur la fédération OIDC :

- Utilisez OpenID Connect dans vos GitHub flux de travail en configurant OpenID Connect dans Amazon Web Services
- Amazon Cognito Identity dans le Guide Amplify Libraries for Android et Amazon Cognito Identity dans le Guide Amplify Libraries for Swift.
- Comment utiliser un identifiant externe pour autoriser l'accès à vos AWS ressources sur le blog de AWS sécurité fournit des conseils sur la configuration sécurisée de l'accès entre comptes et de la fédération d'identités externes.

Création d'un fournisseur d'identité OpenID Connect (OIDC) dans IAM

Les fournisseurs d'identité OIDC IAM sont des entités qui, dans IAM, décrivent un service de fournisseur d'identité (IdP) prenant en charge la norme OpenID Connect (OIDC), comme Google ou Salesforce. Vous utilisez un fournisseur d'identité IAM OIDC lorsque vous souhaitez établir la confiance entre un OIDC-compatible IdP et votre. Compte AWS Cela est utile lorsque vous créez une application mobile ou une application Web qui nécessite l'accès à AWS des ressources, mais que vous ne souhaitez pas créer de code de connexion personnalisé ni gérer vos propres identités d'utilisateur. Pour plus d'informations sur ce scénario, consultez the section called "Fédération OIDC".

Vous pouvez créer et gérer un fournisseur d'identité IAM OIDC à l' Console de gestion AWS aide des outils pour Windows ou de l' PowerShellAPI IAM. AWS Command Line Interface

Après avoir créé un fournisseur d'identité OIDC IAM, vous devez créer un ou plusieurs rôles IAM. Un rôle est une identité AWS qui ne possède pas ses propres informations d'identification (comme le fait un utilisateur). Mais dans ce contexte, un rôle est attribué dynamiquement à un principal fédéré OIDC qui est authentifié par le fournisseur d'identité (IdP) de votre organisation. Le rôle permet à l'IdP de votre organisation de demander des informations d'identification de sécurité temporaires pour accéder à AWS. Les politiques attribuées au rôle déterminent ce que les utilisateurs sont autorisés à faire dans AWS. Pour créer un rôle pour un fournisseur d'identité tiers, consultez la section Créer un rôle pour un fournisseur d'identité tiers.

> **Important**
>
> Lorsque vous configurez des politiques IAM basées sur l'identité pour les actions qui prennent en charge les ressources `oidc-provider`, IAM évalue l'URL complète du fournisseur d'identité OIDC, y compris les chemins spécifiés. Si l'URL de votre fournisseur d'identité OIDC comporte un chemin, vous devez inclure ce chemin dans l'ARN `oidc-provider` en tant que valeur de l'élément `Resource`. Vous avez également la possibilité d'ajouter une barre oblique et un caractère générique (`/*`) au domaine de l'URL ou d'utiliser des caractères génériques (`*` et `?`) à n'importe quel endroit du chemin de l'URL. Si l'URL du fournisseur d'identité OIDC dans la demande ne correspond pas à la valeur définie dans l'élément `Resource` de la politique, la demande échoue.

Pour résoudre les problèmes courants liés à la fédération OIDC IAM, voir Résoudre les erreurs liées à l'OIDC sur Re:POST. AWS

Rubriques

- Conditions préalables : valider la configuration de votre fournisseur d'identité
- Création et gestion d'un fournisseur OIDC (console)
- Création et gestion d'un fournisseur d'identité IAM OIDC (AWS CLI)
- Création et gestion d'un fournisseur d'identité OIDC (AWS API)

Conditions préalables : valider la configuration de votre fournisseur d'identité

Avant de créer un fournisseur d'identité IAM OIDC, vous devez obtenir les informations suivantes auprès de votre IdP. Pour plus d'informations sur l'obtention des informations de configuration du fournisseur OIDC, consultez la documentation de votre IdP.

1. Déterminez l'URL publique de votre fournisseur d'identité OIDC. L'URL doit commencer https://. par Selon la norme OIDC, les composants de chemin sont autorisés mais pas les paramètres de requête. Généralement, l'URL se compose uniquement d'un nom d'hôte, comme https://server.example.org ou https://example.com. L'URL ne doit pas contenir de numéro de port.
2. Ajoutez /.well- known/openid -configuration à la fin de l'URL de votre fournisseur d'identité OIDC pour voir le document de configuration et les métadonnées du fournisseur accessibles au public. Vous devez disposer d'un document de découverte au format JSON avec le document de configuration du fournisseur et les métadonnées qui peuvent être récupérées à partir de l'URL du point de terminaison de découverte du fournisseur OpenID Connect.
3. Confirmez que les valeurs suivantes sont incluses dans les informations de configuration de votre fournisseur. Si l'un de ces champs est absent de votre configuration OpenID, vous devez mettre à jour votre document de découverte. Ce processus peut varier en fonction de votre fournisseur d'identité, il convient donc de suivre la documentation de votre IdP pour mener à bien cette tâche.

   - émetteur : l'URL de votre domaine.
   - jwks_uri : point de terminaison JSON Web Key Set (JWKS) auprès duquel IAM obtient vos clés publiques. Votre fournisseur d'identité doit inclure un point de terminaison JSON Web Key Set (JWKS) dans la configuration OpenID. Cette URI définit où obtenir les clés publiques utilisées pour vérifier les jetons signés auprès de votre fournisseur d'identité.

   > **Note**
   >
   > Le JSON Web Key Set (JWKS) doit contenir au moins une clé et peut comporter au maximum 100 clés RSA et 100 clés EC. Si le JWKS de votre

fournisseur d'identité OIDC contient plus de 100 clés RSA ou 100 clés EC, une `InvalidIdentityToken` exception sera renvoyée lors de l'utilisation de l'opération d'AssumeRoleWithWebIdentityAPI avec un JWT signé par un type de clé dépassant la limite de 100 clés. Par exemple, si un JWT est signé avec l'algorithme RSA et qu'il y a plus de 100 clés RSA dans le JWKS de votre fournisseur, une exception `InvalidIdentityToken` sera renvoyée.

- claims_supported : informations sur l'utilisateur qui vous permettent de vous assurer que les réponses d'authentification OIDC de votre IdP contiennent les attributs requis AWS utilisés dans les politiques IAM pour vérifier les autorisations des principaux fédérés OIDC. Pour obtenir la liste des clés de condition IAM pouvant être utilisées pour les demandes, consultez Clés disponibles pour AWS Fédération OIDC.
  - aud : vous devez déterminer la valeur de la demande de public que votre IdP émet dans les jetons Web JSON (JWT). La demande de public (aud) est spécifique à l'application et identifie les destinataires prévus du jeton. Lorsque vous enregistrez une application mobile ou Web auprès d'un fournisseur OpenID Connect, celui-ci établit un identifiant client qui identifie l'application. L'ID client est un identifiant unique pour votre application. Il est transmis dans la demande aud pour l'authentification. La demande aud doit correspondre à la valeur Public lors de la création de votre fournisseur d'identité IAM OIDC.
  - iat : les réclamations doivent inclure une valeur pour `iat` qui représente l'heure à laquelle le jeton d'identification est émis.
  - iss : l'URL du fournisseur d'identité. L'URL doit commencer par https://et correspondre à l'URL du fournisseur fourni à IAM. Conformément à la norme OIDC, les composants du chemin sont autorisés, mais pas les paramètres de requête. Généralement, l'URL se compose uniquement d'un nom d'hôte, comme https://server.example.org ou https://example.com. L'URL ne doit pas contenir de numéro de port.
- response_types_supported: id_token
- subject_types_supported: public
- id_token_signing_alg_values_supported: RS256, RS384, RS512, ES256, ES384, ES512

> **Note**
>
> Vous pouvez inclure des revendications supplémentaires comme `my_custom_claim` dans l'exemple ci-dessous ; toutefois, la réclamation AWS STS sera ignorée.

```
{
  "issuer": "https://example-domain.com",
  "jwks_uri": "https://example-domain.com/jwks/keys",
  "claims_supported": [
    "aud",
    "iat",
    "iss",
    "name",
    "sub",
    "my_custom_claim"
  ],
  "response_types_supported": [
    "id_token"
  ],
  "id_token_signing_alg_values_supported": [
    "RS256",
    "RS384",
    "RS512",
    "ES256",
    "ES384",
    "ES512"
  ],
  "subject_types_supported": [
    "public"
  ]
}
```

Création et gestion d'un fournisseur OIDC (console)

Observez les instructions suivantes pour créer et gérer un fournisseur d'identité OIDC IAM dans la Console de gestion AWS.

> **Important**
>
> Si vous utilisez un fournisseur d'identité OIDC de Google, Facebook ou Amazon Cognito, ne créez pas de fournisseur d'identité IAM distinct selon cette procédure. Ces fournisseurs d'identité OIDC sont déjà intégrés AWS et sont disponibles pour votre utilisation. Au lieu de cela, créez de nouveaux rôles pour votre fournisseur d'identité, en consultant Création d'un rôle pour la fédération OpenID Connect (console).

Pour créer un fournisseur d'identité OIDC IAM (console)

1. Avant de créer un fournisseur d'identité OIDC IAM, vous devez enregistrer votre application auprès du fournisseur d'identité afin de recevoir un ID client. L'ID client (également appelé public ciblé) est un identifiant unique pour votre application. Il est émis lorsque vous enregistrez l'application auprès du fournisseur d'identité. Pour plus d'informations sur l'obtention d'un ID client, consultez la documentation de votre fournisseur d'identité.

   > **Note**
   >
   > AWS sécurise la communication avec les fournisseurs d'identité OIDC (IdPs) à l'aide de notre bibliothèque d'autorités de certification (CA) racines fiables pour vérifier le certificat TLS du point de terminaison JSON Web Key Set (JWKS). Si votre IdP OIDC repose sur un certificat qui n'est pas signé par l'une de ces autorités de certification fiables, nous sécurisons la communication à l'aide des empreintes définies dans la configuration de l'IdP. AWS reviendra à la vérification des empreintes digitales si nous ne parvenons pas à récupérer le certificat TLS ou si TLS v1.3 est requis.

2. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
3. Dans le panneau de navigation, sélectionnez Fournisseurs d'identité, puis Ajouter un fournisseur.
4. Pour Configurer le fournisseur, sélectionnez OpenID Connect.
5. Sous URL du fournisseur, tapez l'URL du fournisseur d'identité. L'URL doit respecter les restrictions suivantes :

   - L'URL est sensible à la casse.
   - L'URL doit commencer par **`https://`**.
   - L'URL ne doit pas contenir de numéro de port.
   - Au sein de votre Compte AWS, chaque fournisseur d'identité IAM OIDC doit utiliser une URL unique. Si vous essayez de soumettre une URL qui a déjà été utilisée pour un fournisseur OpenID Connect dans le Compte AWS, vous obtiendrez un message d'erreur.

6. Pour Audience, saisissez l'ID client de l'application que vous avez enregistrée auprès de l'IdP et que vous avez reçueStep 1, et à AWS laquelle vous envoyez des demandes. Si vous disposez d'ID client supplémentaires (également appelés publics ciblés) pour ce fournisseur d'identité, vous pouvez les ajouter ultérieurement sur la page de détails du fournisseur.

> **Note**
>
> Si votre jeton JWT IdP inclut la demande `azp`, saisissez cette valeur comme valeur Public.
> Si votre fournisseur d'identité OIDC définit à la fois le jeton `aud` et le `azp` réclame, AWS STS il utilisera la valeur de la `azp` réclamation comme `aud` réclamation.

7. (Facultatif) Pour Ajouter des balises, vous pouvez ajouter des paires clé-valeur pour vous aider à identifier et à organiser vos. IdPs Vous pouvez également utiliser des balises pour contrôler l'accès aux ressources AWS . Pour en savoir plus sur le balisage des fournisseurs d'identité OIDC IAM, reportez-vous à la section Balisage de fournisseurs d'identité OpenID Connect (OIDC). Choisissez Ajouter une balise. Saisissez des valeurs pour chaque paire clé-valeur de balise.
8. Vérifiez les informations que vous avez fournies. Lorsque vous avez terminé, sélectionnez Ajouter un fournisseur. IAM tentera de récupérer et d'utiliser l'empreinte numérique de l'autorité de certification intermédiaire supérieure du certificat du serveur IdP OIDC pour créer le fournisseur d'identité IAM OIDC.

> **Note**
>
> La chaîne de certificats du fournisseur d'identité OIDC doit commencer par l'URL du domaine ou de l'émetteur, puis par le certificat intermédiaire et enfin par le certificat racine. Si l'ordre de la chaîne de certificats est différent ou comprend des certificats en double ou supplémentaires, vous recevez une erreur de non-concordance de signature et STS ne parvient pas à valider le jeton Web JSON (JWT). Corrigez l'ordre des certificats dans la chaîne renvoyée par le serveur pour résoudre l'erreur. Pour plus d'informations sur les normes relatives aux chaînes de certificats, consultez certificate_list dans la RFC 5246 sur le site Web de la série RFC.

9. Attribuez un rôle IAM à votre fournisseur d'identité pour autoriser les identités des utilisateurs externes gérées par votre fournisseur d'identité à accéder aux AWS ressources de votre compte. Pour en savoir plus sur la création de rôles pour la fédération d'identité, consultez la section Créer un rôle pour un fournisseur d'identité tiers.

> **Note**
>
> L'OIDC IdPs utilisé dans une politique de confiance des rôles doit se trouver dans le même compte que le rôle qui l'approuve.

Pour ajouter ou supprimer une empreinte pour un fournisseur d'identité OIDC IAM (console)

> **Note**
>
> AWS sécurise la communication avec les fournisseurs d'identité OIDC (IdPs) à l'aide de notre bibliothèque d'autorités de certification (CA) racines fiables pour vérifier le certificat TLS du point de terminaison JSON Web Key Set (JWKS). Si votre IdP OIDC repose sur un certificat qui n'est pas signé par l'une de ces autorités de certification fiables, nous sécurisons la communication à l'aide des empreintes définies dans la configuration de l'IdP. AWS reviendra à la vérification des empreintes digitales si nous ne parvenons pas à récupérer le certificat TLS ou si TLS v1.3 est requis.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Fournisseurs d'identité. Sélectionnez ensuite le nom du fournisseur d'identité IAM que vous souhaitez mettre à jour.
3. Choisissez l'onglet Vérification du point de terminaison, puis dans la section Empreintes numériques, sélectionnez Gérer. Pour saisir une nouvelle valeur d'empreinte, sélectionnez Ajouter une empreinte. Pour supprimer une empreinte numérique, sélectionnez Supprimer en regard de l'empreinte que vous souhaitez supprimer.

   > **Note**
   >
   > Un fournisseur d'identité OIDC IAM peut avoir entre 1 (minimum) et 5 empreintes.

   Lorsque vous avez terminé, sélectionnez Enregistrer les modifications.

Pour ajouter une audience pour un fournisseur d'identité OIDC IAM (console)

1. Dans le panneau de navigation, sélectionnez Identity providers (Fournisseurs d'identité), puis le nom du fournisseur d'identité IAM que vous voulez mettre à jour.
2. Dans la section Audiences, sélectionnez Actions et Ajouter une audience.
3. Tapez l'ID client de l'application que vous avez enregistrée auprès de l'IdP et que vous avez reçueStep 1, et à laquelle vous adresserez des demandes. AWS Sélectionnez ensuite Ajouter des audiences.

> **Note**
>
> Un fournisseur d'identité OIDC IAM peut avoir entre 1 (minimum) et 100 audiences (maximum).

Pour supprimer une audience pour un fournisseur d'identité OIDC IAM (console)

1. Dans le panneau de navigation, sélectionnez Identity providers (Fournisseurs d'identité), puis le nom du fournisseur d'identité IAM que vous voulez mettre à jour.
2. Dans la section Audiences, activez la case d'option en regard de l'audience que vous souhaitez supprimer, puis sélectionnez Actions.
3. Sélectionnez Supprimer l'audience. Une nouvelle fenêtre s'ouvre.
4. Si vous supprimez une audience, les identités fédérées avec l'audience ne peuvent pas endosser les rôles associés à l'audience. Dans la fenêtre, lisez le message d'avertissement et confirmez que vous souhaitez supprimer l'audience en saisissant le mot `remove` dans le champ.
5. Sélectionnez Supprimer pour supprimer l'audience.

Pour supprimer un fournisseur d'identité OIDC IAM (console)

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Identity providers (Fournisseurs d'identité).
3. Sélectionnez la case à cocher en regard du fournisseur d'identité IAM que vous souhaitez supprimer. Une nouvelle fenêtre s'ouvre.
4. Confirmez que vous souhaitez supprimer le fournisseur en saisissant le mot `delete` dans le champ. Ensuite, choisissez Supprimer.

Création et gestion d'un fournisseur d'identité IAM OIDC (AWS CLI)

Vous pouvez utiliser les AWS CLI commandes suivantes pour créer et gérer des fournisseurs d'identité IAM OIDC.

Pour créer un fournisseur d'identité IAM OIDC (AWS CLI)

1. (Facultatif) Pour obtenir la liste de tous les fournisseurs d'identité OIDC IAM de votre compte AWS , exécutez la commande suivante :
   - `aws iam list-open-id-connect-providers`
2. Pour créer un fournisseur d'identité OIDC IAM, exécutez la commande suivante :
   - `aws iam create-open-id-connect-provider`

Pour mettre à jour la liste des empreintes numériques des certificats de serveur pour un fournisseur d'identité IAM OIDC existant (AWS CLI)

- Pour mettre à jour la liste des empreintes de certificat de serveur pour un fournisseur d'identité OIDC IAM, exécutez la commande suivante :
  - `aws iam update-open-id-connect-provider-thumbprint`

Pour baliser un fournisseur d'identité IAM OIDC existant (AWS CLI)

- Pour baliser un fournisseur d'identité OIDC IAM existant, exécutez la commande suivante :
  - `aws iam tag-open-id-connect-provider`

Pour répertorier les balises d'un fournisseur d'identité IAM OIDC existant (AWS CLI)

- Pour afficher la liste des balises d'un fournisseur d'identité OIDC IAM existant, exécutez la commande suivante :
  - `aws iam list-open-id-connect-provider-tags`

Pour supprimer des balises sur un fournisseur d'identité IAM OIDC (AWS CLI)

- Pour supprimer les balises d'un fournisseur d'identité OIDC IAM existant, exécutez la commande suivante :
  - `aws iam untag-open-id-connect-provider`

Pour ajouter ou supprimer un ID client à un fournisseur d'identité IAM OIDC existant (AWS CLI)

1. (Facultatif) Pour obtenir la liste de tous les fournisseurs d'identité OIDC IAM de votre compte AWS , exécutez la commande suivante :
   - `aws iam list-open-id-connect-providers`
2. (Facultatif) Pour obtenir des informations détaillées sur un fournisseur d'identité OIDC IAM, exécutez la commande suivante :
   - `aws iam get-open-id-connect-provider`
3. Pour ajouter un nouvel ID client à un fournisseur d'identité OIDC IAM existant, exécutez la commande suivante :
   - `aws iam add-client-id-to-open-id-connect-provider`
4. Pour supprimer un client d'un fournisseur d'identité OIDC IAM existant, exécutez la commande suivante :
   - `aws iam remove-client-id-from-open-id-connect-provider`

Pour supprimer un fournisseur d'identité IAM OIDC (AWS CLI)

1. (Facultatif) Pour obtenir la liste de tous les fournisseurs d'identité OIDC IAM de votre compte AWS , exécutez la commande suivante :
   - `aws iam list-open-id-connect-providers`
2. (Facultatif) Pour obtenir des informations détaillées sur un fournisseur d'identité OIDC IAM, exécutez la commande suivante :
   - `aws iam get-open-id-connect-provider`
3. Pour supprimer un fournisseur d'identité OIDC IAM, exécutez la commande suivante :

- `aws iam delete-open-id-connect-provider`

Création et gestion d'un fournisseur d'identité OIDC (AWS API)

Vous pouvez utiliser les commandes de l'API IAM suivantes pour créer et gérer des fournisseurs OICD.

Pour créer un fournisseur d'identité IAM OIDC (AWS API)

1. (Facultatif) Pour obtenir la liste de tous les fournisseurs d'identité OIDC IAM de votre compte AWS , appelez l'opération suivante :
   - `ListOpenIDConnectProviders`
2. Pour créer un fournisseur d'identité OIDC IAM, appelez l'opération suivante :
   - `CreateOpenIDConnectProvider`

Pour mettre à jour la liste des empreintes numériques des certificats de serveur pour un fournisseur d'identité IAM OIDC existant (AWS API)

- Pour mettre à jour la liste des empreintes de certificat de serveur pour un fournisseur d'identité OIDC IAM, appelez l'opération suivante :
  - `UpdateOpenIDConnectProviderThumbprint`

Pour baliser un fournisseur d'identité IAM OIDC existant (AWS API)

- Pour baliser un fournisseur d'identité OIDC IAM existant, appelez l'opération suivante :
  - `TagOpenIDConnectProvider`

Pour répertorier les balises d'un fournisseur d'identité IAM OIDC existant (AWS API)

- Pour afficher la liste des balises d'un fournisseur d'identité OIDC IAM existant, appelez l'opération suivante :
  - `ListOpenIDConnectProviderTags`

Pour supprimer les balises d'un fournisseur d'identité IAM OIDC existant (AWS API)

- Pour supprimer les balises d'un fournisseur d'identité OIDC IAM existant, appelez l'opération suivante :
  - `UntagOpenIDConnectProvider`

Pour ajouter ou supprimer un ID client à un fournisseur d'identité IAM OIDC existant (AWS API)

1. (Facultatif) Pour obtenir la liste de tous les fournisseurs d'identité OIDC IAM de votre compte AWS , appelez l'opération suivante :
   - `ListOpenIDConnectProviders`
2. (Facultatif) Pour obtenir des informations détaillées sur un fournisseur d'identité OIDC IAM, appelez l'opération suivante :
   - `GetOpenIDConnectProvider`
3. Pour ajouter un nouvel ID client à un fournisseur d'identité OIDC IAM existant, appelez l'opération suivante :
   - `AddClientIDToOpenIDConnectProvider`
4. Pour supprimer un ID client d'un fournisseur d'identité OIDC IAM existant, appelez l'opération suivante :
   - `RemoveClientIDFromOpenIDConnectProvider`

Pour supprimer un fournisseur d'identité IAM OIDC (AWS API)

1. (Facultatif) Pour obtenir la liste de tous les fournisseurs d'identité OIDC IAM de votre compte AWS , appelez l'opération suivante :
   - `ListOpenIDConnectProviders`
2. (Facultatif) Pour obtenir des informations détaillées sur un fournisseur d'identité OIDC IAM, appelez l'opération suivante :
   - `GetOpenIDConnectProvider`
3. Pour supprimer un fournisseur d'identité OIDC IAM, appelez l'opération suivante :

- `DeleteOpenIDConnectProvider`

Obtention de l'empreinte numérique d'un fournisseur d'identité OpenID Connect

Lorsque vous créez un fournisseur d'identité OpenID Connect (OIDC) dans IAM, ce dernier exige l'empreinte de l'autorité de certification intermédiaire supérieure qui a signé le certificat utilisé par le fournisseur d'identité (IdP) externe. L'empreinte numérique est la signature du certificat de l'autorité de certification qui a été utilisé pour délivrer le certificat pour l' OIDC-compatible IdP. Lorsque vous créez un fournisseur d'identité IAM OIDC, vous faites confiance aux identités authentifiées par cet IdP pour accéder à votre. Compte AWS En utilisant l'empreinte numérique du certificat de l'autorité de certification, vous approuvez n'importe quel certificat émis par cette autorité de certification avec le même nom de DNS que celui qui est enregistré. Ainsi, vous n'avez plus besoin de mettre à jour les approbations dans chaque compte lorsque vous renouvelez le certificat de signature de l'IdP.

> **Important**
>
> Dans la plupart des cas, le serveur de fédération utilise deux certificats différents.
>
> - La première établit une connexion HTTPS entre AWS et votre IdP. Elle doit être émise par une autorité de certification racine publique bien connue, telle qu' AWS Certificate Manager. Cela permet au client de vérifier la fiabilité et l'état du certificat.
> - Le second est utilisé pour crypter les jetons et doit être signé par un opérateur privé ou public racine CA.

Vous pouvez créer un fournisseur d'identité IAM OIDC à l'aide AWS Command Line Interface des outils pour Windows ou de l' PowerShellAPI IAM. Lorsque vous utilisez ces méthodes, vous avez la possibilité de fournir manuellement une empreinte numérique. Si vous choisissez de ne pas inclure d'empreinte numérique, IAM récupérera l'empreinte numérique de l'autorité de certification intermédiaire supérieure du certificat du serveur IdP OIDC. Si vous choisissez d'inclure une empreinte numérique, vous devez l'obtenir manuellement et la fournir à AWS.

Lorsque vous créez un fournisseur d'identité OIDC avec la console IAM, IAM tente de récupérer l'empreinte numérique de l'autorité de certification intermédiaire supérieure du certificat du serveur IdP OIDC pour vous.

Nous vous recommandons d'obtenir également l'empreinte numérique de votre IdP OIDC manuellement et de vérifier qu'IAM a récupéré la bonne empreinte numérique. Pour plus d'informations sur l'obtention d'empreintes numériques de certificat, consultez les sections suivantes.

> **Note**
>
> AWS sécurise la communication avec les fournisseurs d'identité OIDC (IdPs) à l'aide de notre bibliothèque d'autorités de certification (CA) racines fiables pour vérifier le certificat TLS du point de terminaison JSON Web Key Set (JWKS). Si votre IdP OIDC repose sur un certificat qui n'est pas signé par l'une de ces autorités de certification fiables, nous sécurisons la communication à l'aide des empreintes définies dans la configuration de l'IdP. AWS reviendra à la vérification des empreintes digitales si nous ne parvenons pas à récupérer le certificat TLS ou si TLS v1.3 est requis.

> **Important**
>
> Si votre fournisseur d'identité OIDC utilise des hôtes différents pour le point de terminaison de découverte (l'URL de l'émetteur) et le point de terminaison JWKS (`jwks_uri`), vous devez inclure l'empreinte numérique des deux points de terminaison dans la liste des empreintes numériques de votre fournisseur OIDC. AWS valide l'empreinte numérique des deux terminaux lors du retour à la vérification des empreintes digitales. Si l'une des empreintes digitales est absente de la liste configurée, l'authentification échoue.
> La plupart des fournisseurs OIDC hébergent les deux terminaux sur le même domaine, de sorte qu'une seule empreinte numérique couvre les deux. Vous n'avez besoin d'ajouter une empreinte numérique distincte que si votre fournisseur utilise des hôtes ou des certificats différents pour les deux terminaux.

Obtention de l'empreinte numérique du certificat

Vous utilisez un navigateur Web et l'outil de ligne de commande OpenSSL pour obtenir l'empreinte numérique du certificat pour un fournisseur OIDC. Toutefois, vous n'avez pas besoin d'obtenir manuellement l'empreinte numérique du certificat pour créer un fournisseur d'identité OIDC. Vous pouvez utiliser la procédure suivante pour obtenir l'empreinte numérique du certificat de votre fournisseur OIDC.

Pour obtenir l'empreinte d'un IdP OIDC

1. Avant d'obtenir l'empreinte pour un IdP OIDC, vous devez obtenir l'outil de ligne de commande OpenSSL. Cet outil vous permet de télécharger la chaîne de certificats de l'IdP OIDC et de fournir une empreinte du certificat final dans la chaîne de certificats. Si vous avez besoin d'installer et de configurer OpenSSL, suivez les instructions des sections Installer OpenSSL et Configurer OpenSSL.
2. Commencez par l'URL de l'IdP OIDC (par exemple, `https://server.example.com`), puis ajoutez `/.well-known/openid-configuration` pour former l'URL du document de configuration de l'IdP, comme suit :

   `https://server.example.com/.well-known/openid-configuration`

   Ouvrez cette URL dans un navigateur Web, en la *`server.example.com`* remplaçant par le nom de votre serveur IdP.
3. Dans le document affiché, utilisez la fonction rechercher de votre navigateur web pour localiser le texte `"jwks_uri"`. Immédiatement après le texte `"jwks_uri"`, vous verrez apparaître deux points (:) suivis d'une URL. Copiez le nom de domaine complet de l'URL. N'incluez pas `https://` ou le chemin d'accès qui suit le domaine de niveau supérieur.

   ```
   {
    "issuer": "https://accounts.example.com",
    "authorization_endpoint": "https://accounts.example.com/o/oauth2/v2/auth",
    "device_authorization_endpoint": "https://oauth2.exampleapis.com/device/code",
    "token_endpoint": "https://oauth2.exampleapis.com/token",
    "userinfo_endpoint": "https://openidconnect.exampleapis.com/v1/userinfo",
    "revocation_endpoint": "https://oauth2.exampleapis.com/revoke",
    "jwks_uri": "https://www.exampleapis.com/oauth2/v3/certs",
   ...
   ```

   > **Note**
   >
   > Si vos points de terminaison Discovery et JWKS utilisent des hôtes différents, notez le nom de domaine complet figurant dans l'URL de votre émetteur (l'URL que vous avez utiliséeStep 2, sans le `/.well-known/openid-configuration` chemin). Si ce domaine est différent du `jwks_uri` domaine que vous avez copié au cours de cette

étape, procédez comme suit Step 6 pour les deux domaines et Step 4 incluez les deux empreintes digitales lors de la création de votre fournisseur OIDC.

4. Utilisez l'outil de ligne de commande OpenSSL pour exécuter la commande suivante. Remplacez-le *keys.example.com* par le nom de domaine dans lequel vous avez obtenuStep 3.

```
openssl s_client -servername keys.example.com -showcerts -
connect keys.example.com:443
```

5. Dans votre fenêtre de commande, faites défiler jusqu'à afficher un certificat similaire à l'exemple suivant. Si plusieurs certificats sont affichés, recherchez le dernier certificat affiché (à la fin de la sortie de commande). Celui-ci contient le certificat de l'autorité de certification (CA) intermédiaire supérieure dans la chaîne d'autorité de certification.

```
-----BEGIN CERTIFICATE-----
 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
 21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
 rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
 nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
 NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
 -----END CERTIFICATE-----
```

Copiez le certificat (notamment les lignes `-----BEGIN CERTIFICATE-----` et `-----END CERTIFICATE-----`) et collez-le dans un fichier texte. Puis enregistrez le fichier sous le nom **`certificate.crt`**.

> **Note**
>
> La chaîne de certificats du fournisseur d'identité OIDC doit commencer par l'URL du domaine ou de l'émetteur, inclure tous les certificats intermédiaires (le cas échéant)

et se terminer par le certificat racine. Si l'ordre de la chaîne de certificats est différent ou comprend des certificats en double ou supplémentaires, vous recevrez une erreur de non-concordance de signature et STS ne parvient pas à valider le jeton Web JSON (JWT). Corrigez l'ordre des certificats dans la chaîne renvoyée par le serveur pour résoudre l'erreur. Pour plus d'informations sur les normes relatives aux chaînes de certificats, consultez certificate_list dans la RFC 5246 sur le site Web de la série RFC.

6. Utilisez l'outil de ligne de commande OpenSSL pour exécuter la commande suivante.

```
openssl x509 -in certificate.crt -fingerprint -sha1 -noout
```

Votre fenêtre de commande affiche l'empreinte du certificat qui semble similaire à l'exemple suivant :

```
SHA1 Fingerprint=99:0F:41:93:97:2F:2B:EC:F1:2D:DE:DA:52:37:F9:C9:52:F2:0D:9E
```

Supprimez les deux points (:) de cette chaîne pour générer l'empreinte finale, comme ceci :

```
990F4193972F2BECF12DDEDA5237F9C952F20D9E
```

> **Note**
>
> Si votre point de terminaison de découverte utilise un hôte différent de votre point de terminaison JWKS, recommencez Step 6 en Step 4 utilisant le domaine de point de terminaison de découverte (le domaine URL de votre émetteur). Incluez les deux empreintes digitales lors de la création ou de la mise à jour de votre fournisseur d'identité OIDC à l'aide de ou. CreateOpenIDConnectProvider UpdateOpenIDConnectProviderThumbprint

7. Si vous créez le fournisseur d'identité IAM OIDC à l' AWS CLI aide de Tools for Windows ou de l'API IAM PowerShell, la fourniture d'une empreinte numérique est facultative. Si vous choisissez de ne pas inclure d'empreinte numérique lors de la création, IAM récupérera l'empreinte numérique de l'autorité de certification intermédiaire supérieure du certificat du serveur IdP OIDC. Une fois le fournisseur d'identité IAM OIDC créé, vous pouvez comparer cette empreinte numérique à celle récupérée par IAM.

Si vous créez un fournisseur d'identité IAM OIDC dans la console IAM, cette dernière tente de récupérer l'empreinte numérique de l'autorité de certification intermédiaire supérieure du certificat du serveur IdP OIDC pour vous. Vous pouvez comparer cette empreinte numérique à celle récupérée par IAM. Une fois le fournisseur d'identité IAM OIDC créé, vous pouvez consulter son empreinte numérique dans l'onglet Vérification du point de terminaison de la page de la console Récapitulatif du fournisseur OIDC.

> **Important**
>
> Si l'empreinte numérique que vous avez obtenue ne correspond pas à celle que vous voyez dans les détails de l'empreinte numérique du fournisseur d'identité IAM OIDC, vous ne devez pas utiliser le fournisseur OIDC. À la place, supprimez le fournisseur OIDC créé et réessayez de créer le fournisseur OIDC après un certain temps. Vérifiez que les empreintes numériques correspondent avant d'utiliser le fournisseur. Si les empreintes ne correspondent toujours pas après une seconde tentative, visitez le Forum IAM pour contacter AWS.

Installer OpenSSL

Si OpenSSL n'est pas déjà installée, suivez les instructions de cette section.

Pour installer OpenSSL sous Linux ou Unix

1. Accédez à OpenSSL : Source, Tarballs (). https://openssl.org/source/
2. Téléchargez la source la plus récente et créez le package.

Pour installer OpenSSL sous Windows

1. Accédez à OpenSSL : Binary Distributions (https://wiki.openssl.org/index.php/Binaries) pour obtenir la liste des sites à partir desquels vous pouvez installer la version Windows.
2. Suivez les instructions sur le site que vous avez sélectionné pour démarrer l'installation.
3. Si vous êtes invité à installer le package Microsoft Visual C++ 2008 Redistributable et qu'il n'est pas déjà installé sur votre système, choisissez le lien de téléchargement correspondant à votre environnement. Suivez les instructions fournies par l'Assistant d'installation de Microsoft Visual C ++ 2008 Redistributable.

> **Note**
>
> Si vous n'êtes pas sûr que Microsoft Visual C++ 2008 Redistributable soit déjà installé sur votre système, vous pouvez essayer d'installer d'abord OpenSSL. Le programme d'installation d'OpenSSL affiche une alerte si Microsoft Visual C++ 2008 Redistributable n'est pas encore installé. Veillez à installer l'architecture (32 ou 64 bits) correspondant à la version d'OpenSSL que vous installez.

4. Après avoir installé le package Microsoft Visual C++ 2008 Redistributable, sélectionnez la version des fichiers binaires OpenSSL correspondant à votre environnement et enregistrez le fichier localement. Démarrer l'Assistant d'installation d'OpenSSL.
5. Suivez les instructions décrites dans l'Assistant d'installation d'OpenSSL.

Configurer OpenSSL

Avant d'utiliser les commandes OpenSSL, vous devez configurer le système d'exploitation afin qu'il sache où OpenSSL est installé.

Pour configurer OpenSSL sous Linux ou Unix

1. Sur la ligne de commande, définissez la variable `OpenSSL_HOME` à l'emplacement d'installation d'OpenSSL :

```
$ export OpenSSL_HOME=path_to_your_OpenSSL_installation
```

2. Définissez le chemin d'accès de sorte à inclure l'installation d'OpenSSL :

```
$ export PATH=$PATH:$OpenSSL_HOME/bin
```

> **Note**
>
> Les modifications apportées aux variables d'environnement à l'aide de la commande `export` sont valides uniquement pour la session actuelle. Vous pouvez apporter des modifications persistantes aux variables d'environnement en les définissant dans votre fichier de configuration de shell. Pour plus d'informations, consultez la documentation de votre système d'exploitation.

Pour configurer OpenSSL sous Windows

1. Ouvrez une fenêtre d'invite de commande.
2. Définissez la variable `OpenSSL_HOME` à l'emplacement d'installation d'OpenSSL :

```
C:\> set OpenSSL_HOME=path_to_your_OpenSSL_installation
```

3. Définissez la variable `OpenSSL_CONF` à l'emplacement du fichier de configuration dans votre installation d'OpenSSL :

```
C:\> set OpenSSL_CONF=path_to_your_OpenSSL_installation\bin\openssl.cfg
```

4. Définissez le chemin d'accès de sorte à inclure l'installation d'OpenSSL :

```
C:\> set Path=%Path%;%OpenSSL_HOME%\bin
```

> **Note**
>
> Toutes les modifications que vous apportez aux variables d'environnement Windows dans une fenêtre d'invite de commandes ne sont valides que pour la session de ligne de commande en cours. Vous pouvez apporter des modifications persistantes aux variables d'environnement en les définissant comme propriétés système. Les procédures précises dépendent de la version de Windows que vous utilisez. (Par exemple, dans Windows 7, ouvrez le Panneau de configuration, puis Système et sécurité, Système. Ensuite, choisissez Paramètres système avancés, Avancés, Variables d'environnement.) Pour de plus amples informations, veuillez consulter la documentation Windows.

Identity-provider contrôles pour les fournisseurs OIDC partagés

Pour les fournisseurs d'identité OpenID Connect (OIDC) partagés reconnus (IdPs), IAM exige une évaluation explicite des revendications spécifiques dans les politiques de confiance en matière de rôles. Ces demandes obligatoires, appelées contrôles des fournisseurs d'identité, sont évaluées par IAM lors de la création des rôles et des mises à jour des politiques de confiance. Si la politique de confiance des rôles n'évalue pas les contrôles requis par l'IdP OIDC partagé, la création ou la mise à jour du rôle échouera. Cela garantit que seules les identités autorisées de l'organisation concernée peuvent assumer des rôles et accéder aux ressources AWS . Ce contrôle de sécurité est crucial lorsque les fournisseurs OIDC sont partagés entre plusieurs clients AWS .

Identity-provider les contrôles ne seront pas évalués par IAM pour les politiques de confiance des rôles OIDC existantes. Pour toute modification apportée à la politique de confiance des rôles pour les rôles OIDC existants, IAM exigera que les contrôles du fournisseur d’identité soient inclus dans la politique de confiance des rôles.

Types de fournisseurs OIDC

IAM classe les fournisseurs d’identité OIDC en deux types distincts : privés et partagés. Un IdP OIDC privé peut être détenu et géré par une seule organisation ou peut être un locataire d’un fournisseur SaaS, son URL d’émetteur OIDC servant d’identifiant unique spécifique à cette organisation. En revanche, un IdP OIDC partagé est utilisé par plusieurs organisations, où l’URL de l’émetteur OIDC peut être identique pour toutes les organisations utilisant ce fournisseur d’identité partagé.

Le tableau ci-dessous décrit les principales différences entre les fournisseurs OIDC privés et partagés :

Caractéristiques	Fournisseur OIDC privé	Fournisseur OIDC partagé
Emetteur	Unique à l’organisation	Partagé entre plusieurs organisations
Informations de location	Communiquées par l’intermé diaire d’un émetteur unique	Communiquées par le biais de réclamations dans JWT
Exigences en matière de politique de confiance	Aucune évaluation de demande spécifique requise	Évaluation des demandes spécifiques requise

Fournisseurs d’identité OIDC partagés avec contrôles des fournisseurs d’identité

Lorsque vous créez ou modifiez un fournisseur OIDC dans IAM, le système identifie et évalue automatiquement les revendications requises pour les fournisseurs OIDC partagés reconnus. Si les contrôles du fournisseur d'identité ne sont pas configurés dans la politique de confiance des rôles, la création ou la mise à jour des rôles échouera avec une MalformedPolicyDocument erreur.

Le tableau suivant répertorie les fournisseurs OIDC partagés qui nécessitent des contrôles de fournisseur d’identité dans les politiques de confiance des rôles, ainsi que des informations supplémentaires pour vous aider à configurer les contrôles de fournisseur d’identité.

IdP OIDC	URL OIDC	Demande de location	Demandes requises
Amazon Cognito	`cognito-identity.amazonaws.com`	aud	`cognito-identity.amazonaws.com:aud`
Azure Sentinel	`https://sts.windows.net/33e01921-4d64-4f8c-a055-5bdaffd5e33d`	`sts:RoleSessionName`	`sts:RoleSessionName`
Buildkite	`https://agent.buildkite.com`	sub	`agent.buildkite.com:sub`
Codefresh SaaS	`https://oidc.codefresh.io`	sub	`oidc.codefresh.io:sub`
DVC Studio	`https://studio.datachain.ai/api`	sub	`studio.datachain.ai/api:sub`

GitHub actions	`https://token.actions.githubusercontent.com`	sub ou job_workflow_ref	`token.actions.githubusercontent.com:sub`
			or
			`token.actions.githubusercontent.com:`

IdP OIDC	URL OIDC	Demande de location	Demandes requises
			`job_workflow_ref`
GitHub diffusion en continu du journal d'audit	`https://oidc-configuration.audit-log.githubusercontent.com`	sub	`oidc-configuration.audit-log.githubusercontent.com:sub`
GitHub vstoken	`https://vstoken.actions.githubusercontent.com`	sub	`vstoken.actions.githubusercontent.com:sub`
GitLab	`https://gitlab.com`	sub	`gitlab.com:sub`

IdP OIDC	URL OIDC	Demande de location	Demandes requises
IBM Turbonomic SaaS*	• https://rh-oidc.s3.us-east-1.amazonaws.com/22ejnvnnturfmt6km08iidd0nt4hekbn7 • https://rh-oidc.s3.us-east-1.amazonaws.com/23e3sd27sju1hoou6ohfs68vbno607tr • https://rh-oidc.s3.us-east-1.amazonaws.com/23ne21h005qjl3n33d8dui5dlrmv2tmg • https://rh-oidc.s3.us-east-1.amazonaws.com/24jrf12m5dj7ljlfb4ta2frhrcoadm26	sub	• rh-oidc.s3.us-east-1.amazonaws.com/22ejnvnnturfmt6km08iidd0nt4hekbn7:sub • rh-oidc.s3.us-east-1.amazonaws.com/23e3sd27sju1hoou6ohfs68vbno607tr:sub • rh-oidc.s3.us-east-1.amazonaws.com/23ne21h005qjl3n33d8dui5dlrmv2tmg:sub • rh-oidc.s3.us-east-1.amazonaws.com/24jrf12m5dj7ljlfb4ta2frhrcoadm26:sub

IdP OIDC	URL OIDC	Demande de location	Demandes requises
	• https://oidc.op1.openshiftapps.com/2f785sojlpb85i7402pk3qogugim5nfb • https://oidc.op1.openshiftapps.com/2c51blsaqa9gkjt0o9rt11mle8mmropu		• oidc.op1.openshiftapps.com/2f785sojlpb85i7402pk3qogugim5nfb:sub • oidc.op1.openshiftapps.com/2c51blsaqa9gkjt0o9rt11mle8mmropu:sub
Pulumi Cloud	https://api.pulumi.com/oidc	aud	api.pulumi.com/oidc:aud
sandboxes.cloud	https://sandboxes.cloud	aud	sandboxes.cloud:aud
Scalr	https://scalr.io	sub	scalr.io:sub
Shisho Cloud	https://tokens.cloud.shisho.dev	sub	tokens.cloud.shisho.dev:sub
Terraform Cloud	https://app.terraform.io	sub	app.terraform.io:sub

IdP OIDC	URL OIDC	Demande de location	Demandes requises
Upbound	`https://proidc.upbound.io`	sub	`proidc.upbound.io:sub`
Point de terminaison global Vercel	`https://oidc.vercel.com`	aud	`oidc.vercel.com:aud`

* IBM Turbonomic met régulièrement à jour l'URL de l'émetteur OIDC avec les nouvelles versions de la plateforme. Nous ajouterons d'autres émetteurs Turbonomic OIDC dans le champ d'application en tant que fournisseur partagé, si nécessaire.

Pour tout nouvel OIDC IdPs identifié par IAM comme étant partagé, les contrôles du fournisseur d'identité requis pour les politiques de confiance en matière de rôles seront documentés et appliqués de la même manière.

Ressources supplémentaires

Ressources supplémentaires :

- Pour en savoir plus sur la création d'un rôle IAM pour une fédération OIDC, consultez Création d'un rôle pour la fédération OpenID Connect (console).
- Pour obtenir la liste des clés de condition IAM pouvant être utilisées pour les demandes, consultez Clés disponibles pour AWS Fédération OIDC.

Fédération SAML 2.0

AWS prend en charge la fédération d'identités avec SAML 2.0 (Security Assertion Markup Language 2.0), une norme ouverte utilisée par de nombreux fournisseurs d'identité (IdPs). Cette fonctionnalité permet l'authentification unique fédérée (SSO), afin que les utilisateurs puissent se connecter aux opérations d' AWS API Console de gestion AWS ou les appeler sans que vous ayez à créer un utilisateur IAM pour tous les membres de votre organisation. En utilisant SAML, vous pouvez simplifier le processus de configuration de la fédération avec AWS, car vous pouvez utiliser le service de l'IdP au lieu d'écrire un code proxy d'identité personnalisé.

> **Note**
>
> La fédération d'identité IAM SAML prend en charge les réponses SAML cryptées provenant de fournisseurs d'identité SAML-based fédérés (). IdPs IAM Identity Center et Amazon Cognito ne prennent pas en charge les assertions SAML chiffrées provenant des fournisseurs d'identité SAML IAM.
>
> Vous pouvez ajouter indirectement la prise en charge des assertions SAML chiffrées à la fédération des groupes d'identités Amazon Cognito avec les groupes d'utilisateurs Amazon Cognito. Les groupes d'utilisateurs disposent d'une fédération SAML indépendante de la fédération SAML IAM et qui prend en charge la signature et le chiffrement SAML. Bien que cette fonctionnalité ne s'étende pas directement aux groupes d'identités, les groupes d'utilisateurs peuvent appartenir IdPs à des groupes d'identités. Pour utiliser le chiffrement SAML avec des groupes d'identités, ajoutez un fournisseur SAML avec chiffrement à un groupe d'utilisateurs qui est un IdP d'un groupe d'identités.
>
> Votre fournisseur SAML doit être en mesure de chiffrer les assertions SAML à l'aide d'une clé fournie par votre groupe d'utilisateurs. Les groupes d'utilisateurs n'accepteront pas les assertions chiffrées à l'aide d'un certificat fourni par IAM.

La fédération IAM prend en charge les cas d'utilisation suivants :

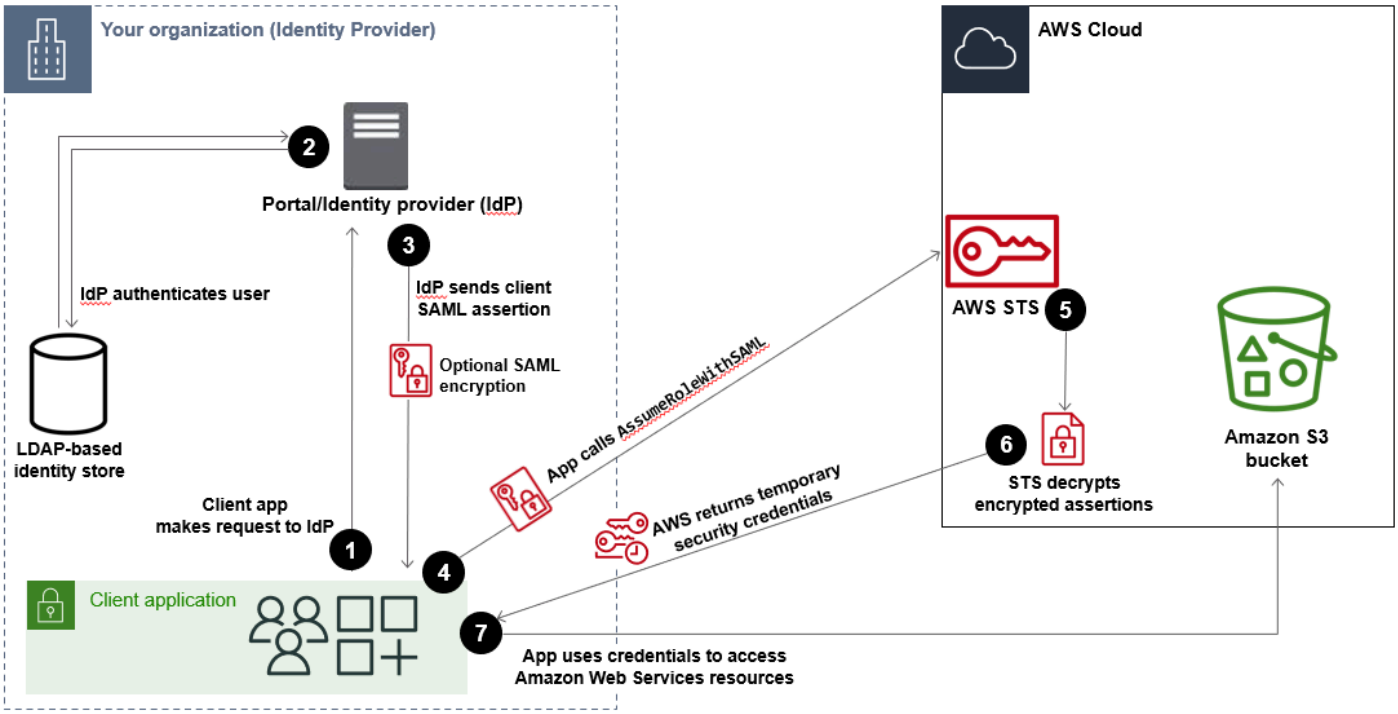
- Accès fédéré pour permettre à un utilisateur ou à une application de votre organisation d'appeler les opérations AWS d'API. Ce cas d'utilisation est traité dans la section suivante. Vous utilisez une assertion SAML (dans le cadre de la réponse d'authentification) qui est générée dans votre organisation pour l'obtention d'informations d'identification temporaires. Ce scénario est similaire à d'autres scénarios de fédération pris en charge par IAM, tels que ceux décrits dans Demande d'identifiants de sécurité temporaires et Fédération OIDC. Cependant, les systèmes SAML 2.0 de votre IdPs organisation gèrent de nombreux détails au moment de l'exécution pour effectuer des vérifications d'authentification et d'autorisation.
- Web-basedauthentification unique (SSO) Console de gestion AWS depuis votre organisation. Les utilisateurs peuvent se connecter à un portail de votre organisation hébergé par un IdP compatible avec SAML 2.0, sélectionner une option à laquelle accéder et être redirigés vers la console sans avoir à fournir d'informations de connexion supplémentaires. AWS Vous pouvez utiliser un IdP SAML tiers pour établir un accès SSO à la console ou créer un IdP personnalisé pour autoriser l'accès de vos utilisateurs externes à la console. Pour plus d'informations sur la création d'un IdP personnalisé, consultez la page Activez l'accès personnalisé du courtier d'identité au AWS console.

Rubriques

- Utilisation SAML-based de la fédération pour accéder à l'API AWS
- Présentation de la configuration de la fédération SAML 2.0
- Vue d'ensemble du rôle pour autoriser SAML-federated l'accès à votre AWS resources
- Identification unique des utilisateurs au sein de SAML-based la fédération
- Création d’un fournisseur d’identité SAML dans IAM
- Configuration de votre IdP SAML 2.0 à l’aide d’une relation d’approbation des parties utilisatrices et ajout de demandes
- Intégrez des fournisseurs de solutions SAML tiers à AWS
- Configuration d’assertions SAML pour la réponse d’authentification
- Permettre aux principaux fédérés SAML 2.0 d'accéder au Console de gestion AWS
- Affichage d’une réponse SAML dans votre navigateur

Utilisation SAML-based de la fédération pour accéder à l'API AWS

Supposons que vous voulez permettre aux employés de copier les données de leurs ordinateurs dans un dossier de sauvegarde. Vous créez une application que les utilisateurs peuvent exécuter sur leurs ordinateurs. Sur l’instance principale, l’application lit et écrit les objets dans un compartiment Amazon S3. Les utilisateurs n'ont pas d'accès direct à AWS. À la place, le processus suivant est utilisé :



1. A l'aide d'une application cliente, un utilisateur de l'organisation demande son authentification par l'IdP de l'organisation.
2. L'IdP authentifie l'utilisateur en le comparant à la base d'identités de l'organisation.
3. L'IdP crée une assertion SAML à l'aide des informations concernant l'utilisateur et l'envoie à l'application client. Lorsque vous activez le chiffrement SAML pour votre IdP SAML IAM, cette assertion est chiffrée par votre IdP externe.
4. L'application cliente appelle l' AWS STS `AssumeRoleWithSAML`API en transmettant l'ARN du fournisseur SAML, l'ARN du rôle à assumer et l'assertion SAML de l'IdP. Si le chiffrement est activé, l'assertion transmise via l'application cliente reste chiffrée pendant le transfert.
5. (Facultatif) AWS STS utilise la clé privée que vous avez chargée depuis votre IdP externe pour déchiffrer l'assertion SAML chiffrée.
6. La réponse de l'API à l'application cliente inclut des informations d'identification de sécurité temporaires.
7. L'application cliente utilise ces informations d'identification de sécurité temporaires pour appeler les opérations d'API Amazon S3.

Présentation de la configuration de la fédération SAML 2.0

Avant de pouvoir utiliser la fédération basée sur SAML 2.0 comme décrit dans le scénario et le schéma précédents, vous devez configurer l'IdP de votre organisation et le vôtre Compte AWS pour qu'ils se fassent mutuellement confiance. La procédure suivante décrit le processus général permettant de configurer cette relation d'approbation. Votre organisation doit utiliser un IdP prenant en charge SAML 2.0, comme Microsoft Active Directory Federation Service (services ADFS qui font partie de Windows Server), Shibboleth, ou tout autre fournisseur compatible avec SAML 2.0.

> **Note**
>
> Pour améliorer la résilience de la fédération, nous vous recommandons de configurer votre IdP et votre fédération AWS pour prendre en charge plusieurs points de terminaison de connexion SAML. Pour plus de détails, consultez l'article du blog sur la AWS sécurité Comment utiliser les points de terminaison SAML régionaux pour le basculement.

Configurez l'IdP de votre organisation et AWS pour se faire confiance

1. Inscrivez-vous AWS en tant que fournisseur de services (SP) auprès de l'IdP de votre organisation. Utilisez le document de métadonnées SAML à partir de `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`

   Pour une liste des *region-code* valeurs possibles, consultez la colonne Région dans les AWS Sign-In points de .

   Vous pouvez éventuellement utiliser le document de métadonnées SAML à partir de`https://signin.aws.amazon.com/static/saml-metadata.xml`.
2. À l'aide de l'IdP de l'organisation, vous générez un fichier XML de métadonnées SAML équivalent, capable de décrire cet IdP en tant que fournisseur d'identité IAM dans AWS. Il doit inclure le nom de l'émetteur, une date de création, une date d'expiration et des clés qui AWS peuvent être utilisées pour valider les réponses d'authentification (assertions) de votre organisation.

   Si vous autorisez l'envoi d'assertions SAML chiffrées depuis votre IdP externe, vous devez générer un fichier de clé privée à l'aide de l'IdP de votre organisation et charger ce fichier dans votre configuration IAM SAML au format de fichier .pem. AWS STS a besoin de cette clé privée pour déchiffrer les réponses SAML qui correspondent à la clé publique chargée sur votre IdP.

> **Note**
>
> Tel que défini par le profil d'interopérabilité V2.0 des métadonnées SAML version 1.0, IAM n'évalue ni ne prend aucune mesure en cas d'expiration des X.509 certificats contenus dans les documents de métadonnées SAML. Si vous êtes préoccupé par l'expiration X.509 des certificats, nous vous recommandons de surveiller les dates d'expiration des certificats et de procéder à une rotation des certificats conformément aux politiques de gouvernance et de sécurité de votre organisation.

3. Dans la console IAM, vous créez un fournisseur d'identité SAML. Au cours du processus, vous chargez le document de métadonnées SAML et la clé de déchiffrement privée générés par l'IdP de votre organisation à l'étape Step 2. Pour de plus amples informations, veuillez consulter Création d'un fournisseur d'identité SAML dans IAM.
4. Dans IAM, créez un ou plusieurs rôles IAM. Dans la politique de confiance du rôle, vous définissez le fournisseur SAML comme principal, ce qui établit une relation de confiance entre votre organisation et AWS. La politique d'autorisation du rôle détermine les actions que les utilisateurs de l'organisation sont autorisés à effectuer dans AWS. Pour de plus amples informations, veuillez consulter Créer un rôle pour un fournisseur d'identité tiers.

   > **Note**
   >
   > Les IdP SAML utilisés dans une politique d'approbation de rôle doivent se trouver dans le même compte que le rôle.

5. Dans l'IdP de l'organisation, vous définissez les assertions qui associent les utilisateurs et les groupes de l'organisation aux rôles IAM. Notez que différents utilisateurs et groupes de l'organisation peuvent être associés à différents rôles IAM. La procédure exacte pour leur mappage dépend de l'IdP utilisé. Dans le scénario précédent d'un dossier Amazon S3 pour les utilisateurs, tous les utilisateurs peuvent être associés au rôle qui fournit les autorisations Amazon S3. Pour de plus amples informations, veuillez consulter Configuration d'assertions SAML pour la réponse d'authentification.

   Si votre IdP active le SSO pour la AWS console, vous pouvez configurer la durée maximale des sessions de console. Pour de plus amples informations, veuillez consulter Permettre aux principaux fédérés SAML 2.0 d'accéder au Console de gestion AWS.

6. Dans l'application que vous créez, vous appelez l' AWS Security Token Service `AssumeRoleWithSAML`API en lui transmettant l'ARN du fournisseur SAML dans lequel vous avez créé[Step 3](), l'ARN du rôle dans lequel vous devez supposer que vous avez créé et l'assertion SAML concernant l'utilisateur actuel que vous obtenez de votre IdP. [Step 4]() AWS s'assure que la demande d'attribution du rôle provient de l'IdP référencé dans le fournisseur SAML.

   Pour plus d'informations, consultez AssumeRoleWithSAML dans la Référence d'API AWS Security Token Service .

7. Si la demande aboutit, l'API retourne des informations d'identification de sécurité temporaires que votre application peut utiliser pour adresser des demandes signées à AWS. Votre application dispose d'informations sur l'utilisateur actuel et peut accéder aux compartiments Amazon S3 spécifiques à celui-ci, comme décrit dans le scénario précédent.

Vue d'ensemble du rôle pour autoriser SAML-federated l'accès à votre AWS resources

Les rôles que vous créez dans IAM définissent ce que les principaux fédérés SAML de votre organisation sont en mesure d'effectuer dans AWS. Lorsque vous créez la politique d'approbation pour le rôle, vous spécifiez le fournisseur d'identité SAML créé précédemment en tant que `Principal`. Vous pouvez également ajouter une `Condition` à la politique d'approbation, afin d'autoriser uniquement les utilisateurs correspondant à certains attributs SAML à accéder au rôle. Par exemple, vous pouvez spécifier que seuls les utilisateurs dont l'affiliation SAML est `staff` (comme indiqué parhttps://openidp.feide.no) sont autorisés à accéder au rôle, comme illustré par l'exemple de politique suivant :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::111122223333:saml-provider/
ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {
      "StringEquals": {
        "saml:aud": "https://us-east-1.signin.aws.amazon.com/saml",
        "saml:iss": "https://openidp.feide.no"
```

```
      },
      "ForAllValues:StringLike": {"saml:edupersonaffiliation": ["staff"]}
    }
  }]
}
```

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws-cn:iam::111122223333:saml-provider/
ExampleOrgSSOProvider"
            },
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "saml:aud": "https://cn-north-1.signin.amazonaws.cn/saml",
                    "saml:iss": "https://openidp.feide.no"
                },
                "ForAllValues:StringLike": {
                    "saml:edupersonaffiliation": [
                        "staff"
                    ]
                }
            }
        }
    ]
}
```

> **Note**
>
> Les IdP SAML utilisés dans une politique d'approbation de rôle doivent se trouver dans le même compte que le rôle.

La clé contextuelle `saml:aud` dans la politique spécifie l'URL que votre navigateur affiche lorsque vous vous connectez à la console. Cette URL de point de terminaison de connexion doit correspondre à l'attribut du destinataire de votre fournisseur d'identité. Vous pouvez inclure des URL de connexion dans des régions spécifiques. AWS recommande d'utiliser des points de terminaison régionaux au lieu du point de terminaison global pour améliorer la résilience de la fédération. Si vous n'avez configuré qu'un seul point de terminaison, vous ne pourrez pas vous fédérer AWS dans le cas peu probable où le point de terminaison deviendrait indisponible. Pour une liste des *region-code* valeurs possibles, consultez la colonne Région dans les AWS Sign-In points de .

L'exemple suivant montre le format de l'URL de connexion avec l'option `region-code`

`https://region-code.signin.aws.amazon.com/saml`

Si le chiffrement SAML est obligatoire, l'URL de connexion doit inclure l'identifiant unique qu' AWS attribue à votre fournisseur SAML, que vous trouverez sur la page de détails du fournisseur d'identité. Dans l'exemple suivant, l'URL de connexion comprend un identifiant unique d'IdP, qui nécessite l'ajout de /acs/ au chemin d'accès de connexion.

`https://region-code.signin.aws.amazon.com/saml/acs/IdP-ID`

Vous spécifiez la politique d'autorisation dans le rôle comme pour tout autre rôle. Par exemple, si les utilisateurs de votre organisation sont autorisés à administrer des instances Amazon Elastic Compute Cloud, vous devez autoriser explicitement les actions Amazon EC2 dans la politique d'autorisations, comme celles figurant dans la politique AmazonEC2FullAccess gérée.

Pour plus d'informations sur les clés SAML que vous pouvez utiliser dans une politique, consultez Clés disponibles pour SAML-based AWS STS fédération.

Identification unique des utilisateurs au sein de SAML-based la fédération

Lors de la création de politiques d'accès dans IAM, il est souvent utile de pouvoir spécifier des autorisations en fonction de l'identité des utilisateurs. Par exemple, dans le cas d'utilisateurs fédérés à l'aide de SAML, une application peut conserver les informations dans Amazon S3 à l'aide d'une structure similaire à celle-ci :

```
amzn-s3-demo-bucket/app1/user1
amzn-s3-demo-bucket/app1/user2
amzn-s3-demo-bucket/app1/user3
```

Vous pouvez créer le compartiment (`amzn-s3-demo-bucket`) et le dossier (`app1`) via la console Amazon S3 ou via le AWS CLI, car il s'agit de valeurs statiques. Cependant, les dossiers spécifiques

à l'utilisateur (*user1user2*,*user3*, etc.) doivent être créés au moment de l'exécution à l'aide de code, car la valeur qui identifie l'utilisateur n'est connue que lors de sa première connexion via le processus de fédération.

Pour créer des politiques qui référencent des détails spécifiques à l'utilisateur dans le nom d'une ressource, l'identité de l'utilisateur doit figurer dans des clés SAML pouvant être utilisées dans les conditions des politiques. Les clés suivantes sont disponibles pour la fédération basée sur SAML 2.0 à utiliser dans les politiques IAM. Vous pouvez utiliser les valeurs retournées par les clés suivantes pour créer des identifiants utilisateur uniques pour des ressources comme des dossiers Amazon S3.

- `AWS`. Valeur de hachage basée sur la concaténation de la valeur de réponse `Issuer` (`saml:iss`) et d'une chaîne avec l'ID de compte `saml:namequalifier` et le nom convivial (dernière partie de l'ARN) du fournisseur SAML dans IAM. La concaténation de l'ID de compte et du nom convivial du fournisseur SAML est disponible dans les politiques IAM en tant que clé `saml:doc`. L'identifiant du compte et le nom du fournisseur doivent être séparés par un «/», comme dans « 123456789012/provider_name ». Pour plus d'informations, reportez-vous à la clé `saml:doc` dans Clés disponibles pour SAML-based AWS STS fédération.

  La combinaison de `NameQualifier` et `Subject` permet d'identifier de manière unique un principal fédéré SAML. L'exemple de pseudo-code suivant montre comment cette valeur est calculée. Ce pseudocode + indique la concaténation, `SHA1` représente une fonction qui produit un résumé de message à l'aide de SHA-1, et `Base64` représente une fonction qui produit une version Base-64 codée de la sortie de hachage.

  ```
  Base64 ( SHA1 ( "https://example.com/saml" + "123456789012" + "/
  MySAMLIdP" ) )
  ```

  Pour plus d'informations sur les clés de politique disponibles pour SAML-based la fédération, consultezClés disponibles pour SAML-based AWS STS fédération.
- `saml:sub` (chaîne). Objet de la demande, qui inclut une valeur qui identifie de manière unique un utilisateur individuel au sein d'une organisation (par exemple, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).
- `saml:sub_type` (chaîne). Cette clé peut être `persistent`, `transient` ou l'URI `Format` complet des éléments `Subject` et `NameID` utilisés dans votre assertion SAML. Une va leur `persistent` indique que la valeur de `saml:sub` reste la même pour l'utilisateur pour toutes les sessions. Si la valeur est `transient`, l'utilisateur a une valeur `saml:sub` différente pour chaque session. Pour plus d'informations sur l'attribut `NameID` de l'élément `Format`, consultez Configuration d'assertions SAML pour la réponse d'authentification.

L'exemple suivant illustre une politique d'autorisation qui utilise les clés précédentes pour accorder des autorisations à un dossier spécifique à un utilisateur dans Amazon S3. La politique suppose que les objets Amazon S3 sont identifiés à l'aide d'un préfixe qui inclut à la fois `saml:namequalifier` et `saml:sub`. Notez que l'élément `Condition` inclut un test pour vérifier que la valeur de `saml:sub_type` est définie sur `persistent`. Si la valeur est `transient`, l'élément `saml:sub` de l'utilisateur peut être différent pour chaque session et vous ne devez pas combiner les valeurs pour identifier des dossiers spécifiques aux utilisateurs.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}",
      "arn:aws:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}/*"
    ],
    "Condition": {"StringEquals": {"saml:sub_type": "persistent"}}
  }
}
```

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
```

```
    ],
    "Resource": [
      "arn:aws-cn:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}",
      "arn:aws-cn:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}/*"
    ],
    "Condition": {"StringEquals": {"saml:sub_type": "persistent"}}
  }
}
```

Pour plus d'informations sur le mappage d'assertions de l'IdP et les clés de politique, consultez Configuration d'assertions SAML pour la réponse d'authentification.

Création d'un fournisseur d'identité SAML dans IAM

Un fournisseur d'identité SAML 2.0 IAM est une entité dans IAM qui décrit un service de fournisseur d'identité (IdP) externe prenant en charge la norme SAML 2.0 (Security Assertion Markup Language 2.0). Vous utilisez un fournisseur d'identité IAM lorsque vous souhaitez établir la confiance entre un SAML-compatible IdP tel que Shibboleth ou Active Directory Federation Services et AWS afin que vos utilisateurs puissent accéder aux ressources. AWS Les fournisseurs d'identité SAML IAM sont utilisés en tant que principaux dans une politique d'approbation IAM.

Pour plus d'informations sur ce scénario, consultez Fédération SAML 2.0.

Vous pouvez créer et gérer un fournisseur d'identité IAM dans Console de gestion AWS ou à l'aide AWS CLI d'Outils pour Windows PowerShell ou d'appels AWS d'API.

Après avoir créé un fournisseur SAML, vous devez créer un ou plusieurs rôles IAM. Un rôle est une identité AWS qui ne possède pas ses propres informations d'identification (comme le fait un utilisateur). Mais dans ce contexte, un rôle est attribué dynamiquement à un principal fédéré SAML qui est authentifié par votre IdP. Le rôle permet à votre IdP de demander des informations d'identification de sécurité temporaires pour accéder à AWS. Les politiques attribuées au rôle déterminent ce que les utilisateurs sont autorisés à faire dans AWS. Pour créer un rôle pour la fédération SAML, consultez Créer un rôle pour un fournisseur d'identité tiers.

Enfin, après avoir créé le rôle, vous complétez l'approbation SAML en configurant votre IdP avec des informations AWS et les rôles que vous souhaitez que vos principaux fédérés SAML utilisent. Il s'agit de la configuration de la relation d'approbation des parties utilisatrices entre votre IdP et AWS.

Pour configurer la relation d'approbation des parties utilisatrices, consultez Configuration de votre IdP SAML 2.0 à l'aide d'une relation d'approbation des parties utilisatrices et ajout de demandes.

Rubriques

- Conditions préalables
- Création et gestion d'un fournisseur d'identité SAML IAM (console)
- Gestion des clés de chiffrement SAML
- Créez et gérez un fournisseur d'identité IAM SAML (AWS CLI)
- Créez et gérez un fournisseur d'identité IAM SAML (AWS API)
- Étapes suivantes

Conditions préalables

Avant de créer un fournisseur d'identité SAML, vous devez obtenir les informations suivantes auprès de votre IdP.

- Obtenez le document de métadonnées SAML auprès de votre IdP. Ce document inclut le nom de l'auteur, des informations d'expiration et des clés qui peuvent être utilisées pour valider les réponses d'authentification (assertions) SAML reçues du fournisseur d'identité (IdP). Pour générer le document de métadonnées, utilisez le logiciel de gestion des identités fournit par votre IdP externe.

  > **Important**
  >
  > Ce fichier de métadonnées inclut le nom de l'auteur, des informations d'expiration et des clés qui peuvent être utilisées pour valider les réponses d'authentification (assertions) SAML reçues du fournisseur d'identité (IdP). Le fichier de métadonnées doit être codé dans un UTF-8 format sans marque d'ordre d'octet (BOM). Pour supprimer la nomenclature, vous pouvez encoder le fichier à l' UTF-8 aide d'un outil d'édition de texte, tel que Notepad ++.
  >
  > Le X.509 certificat inclus dans le document de métadonnées SAML doit utiliser une taille de clé d'au moins 1024 bits. En outre, le X.509 certificat doit également être exempt de toute extension répétée. Vous pouvez utiliser des extensions, mais elles ne peuvent apparaître qu'une seule fois dans le certificat. Si le X.509 certificat ne remplit aucune des conditions, la création de l'IdP échoue et renvoie une erreur « Impossible d'analyser les métadonnées ».

Tel que défini par le profil d'interopérabilité V2.0 des métadonnées SAML version 1.0, IAM n'évalue ni ne prend aucune mesure en cas d'expiration des X.509 certificats contenus dans les documents de métadonnées SAML. Si vous êtes préoccupé par l'expiration X.509 des certificats, nous vous recommandons de surveiller les dates d'expiration des certificats et de procéder à une rotation des certificats conformément aux politiques de gouvernance et de sécurité de votre organisation.

- Lorsque vous choisissez d'activer le chiffrement SAML, vous devez générer un fichier de clé privée à l'aide de votre IdP et charger ce fichier dans votre configuration SAML IAM au format .pem. AWS STS a besoin de cette clé privée pour déchiffrer les réponses SAML qui correspondent à la clé publique chargée vers votre IdP. Les algorithmes suivants sont pris en charge :
  - Algorithmes de chiffrement
    - AES-128
    - AES-256
    - RSA-OAEP
  - Algorithme de transport de clés
    - AES-CBC
    - AES-GCM

  Consultez la documentation de votre fournisseur d'identité pour connaître les étapes à suivre pour générer une clé privée.

  Note

  IAM Identity Center et Amazon Cognito ne prennent pas en charge les assertions SAML chiffrées provenant des fournisseurs d'identité SAML IAM. Vous pouvez ajouter indirectement la prise en charge des assertions SAML chiffrées à la fédération des groupes d'identités Amazon Cognito avec les groupes d'utilisateurs Amazon Cognito. Les groupes d'utilisateurs disposent d'une fédération SAML indépendante de la fédération SAML IAM et qui prend en charge la signature et le chiffrement SAML. Bien que cette fonctionnalité ne s'applique pas directement aux groupes d'identités, les groupes d'utilisateurs peuvent appartenir IdPs à des groupes d'identités. Pour utiliser le chiffrement SAML avec des groupes d'identités, ajoutez un fournisseur SAML avec chiffrement à un groupe d'utilisateurs qui est un IdP d'un groupe d'identités.

Votre fournisseur SAML doit être en mesure de chiffrer les assertions SAML à l'aide d'une clé fournie par votre groupe d'utilisateurs. Les groupes d'utilisateurs n'accepteront pas les assertions chiffrées à l'aide d'un certificat fourni par IAM.

Pour obtenir des instructions sur la configuration de la plupart des éléments disponibles IdPs AWS, notamment sur la manière de générer le document de métadonnées SAML requis, consultezIntégrez des fournisseurs de solutions SAML tiers à AWS.

Pour obtenir de l'aide concernant la fédération SAML, consultez la section Résolution des problèmes liés à la fédération SAML.

Création et gestion d'un fournisseur d'identité SAML IAM (console)

Vous pouvez utiliser le Console de gestion AWS pour créer, mettre à jour et supprimer des fournisseurs d'identité IAM SAML. Pour obtenir de l'aide concernant la fédération SAML, consultez la section Résolution des problèmes liés à la fédération SAML.

Pour créer un fournisseur d'identité SAML IAM (console)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation, sélectionnez Fournisseurs d'identité, puis Ajouter un fournisseur.
3. Pour Configurer le fournisseur, sélectionnez SAML.
4. Saisissez un nom pour le fournisseur d'identité.
5. Pour Document de métadonnées, sélectionnez Choisir un fichier, spécifiez le document de métadonnées SAML que vous avez téléchargé dans the section called "Conditions préalables".

> **Note**
>
> La date limite de validité affichée pour le fournisseur d'identité est définie par IAM au moment de la création du fournisseur. Il ne reflète pas les `cacheDuration` attributs `validUntil` or de votre document de métadonnées SAML. IAM n'utilise pas ces attributs de métadonnées pour déterminer la date limite de validité.
> IAM n'évalue ni ne prend aucune mesure en cas d'expiration du document de métadonnées SAML ou des X.509 certificats contenus dans les documents de métadonnées SAML lors de l'authentification SAML. La validité de l'authentification SAML est appliquée au niveau de l'assertion, à l'aide de l'`NotOnOrAfter`attribut figurant

dans les `SubjectConfirmationData` éléments `Conditions` et de l'assertion. Si vous êtes préoccupé par l'expiration X.509 des certificats, nous vous recommandons de surveiller les dates d'expiration des certificats et de procéder à une rotation des certificats conformément aux politiques de gouvernance et de sécurité de votre organisation.

6. (Facultatif) Pour le chiffrement SAML, choisissez Choisir un fichier, puis sélectionnez le fichier de clé privée que vous avez créé dans the section called "Conditions préalables". Choisissez Exiger le chiffrement pour n'accepter que les requêtes chiffrées provenant de votre IdP.
7. (Facultatif) Pour Ajouter des balises, vous pouvez ajouter des paires clé-valeur pour vous aider à identifier et à organiser vos. IdPs Vous pouvez également utiliser des balises pour contrôler l'accès aux ressources AWS . Pour en savoir plus sur le balisage des fournisseurs d'identité SAML, reportez-vous à la section Balisage de fournisseurs d'identité SAML IAM.

   Choisissez Ajouter une balise. Saisissez des valeurs pour chaque paire clé-valeur de balise.
8. Vérifiez les informations que vous avez fournies. Lorsque vous avez terminé, sélectionnez Ajouter un fournisseur.
9. Attribuez un rôle IAM à votre fournisseur d'identité. Ce rôle donne aux identités d'utilisateurs externes gérées par votre fournisseur d'identité les autorisations nécessaires pour accéder aux AWS ressources de votre compte. Pour en savoir plus sur la création de rôles pour la fédération d'identité, consultez la section Créer un rôle pour un fournisseur d'identité tiers.

> **Note**
>
> Les IdP SAML utilisés dans une politique d'approbation de rôle doivent se trouver dans le même compte que le rôle.

Pour supprimer un fournisseur SAML (console)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation, sélectionnez Fournisseurs d'identité.
3. Activez la case d'option en regard du fournisseur d'identité que vous souhaitez supprimer.
4. Sélectionnez Delete (Supprimer). Une nouvelle fenêtre s'ouvre.

5. Confirmez que vous souhaitez supprimer le fournisseur en saisissant le mot `delete` dans le champ. Ensuite, choisissez Supprimer.

Gestion des clés de chiffrement SAML

Vous pouvez configurer les fournisseurs SAML IAM pour recevoir des assertions chiffrées dans la réponse SAML de votre IdP externe. Les utilisateurs peuvent jouer un rôle dans AWS les assertions SAML cryptées en appelant. `sts:AssumeRoleWithSAML`

Le chiffrement SAML garantit que les assertions sont sécurisées lorsqu'elles passent par des intermédiaires ou des tiers. En outre, cette fonctionnalité vous aide à répondre aux exigences de FedRAMP ou de toute politique de conformité interne qui impose le chiffrement des assertions SAML.

Pour configurer un fournisseur d'identité SAML IAM, consultez Création d'un fournisseur d'identité SAML dans IAM. Pour obtenir de l'aide concernant la fédération SAML, consultez la section Résolution des problèmes liés à la fédération SAML.

Rotation d'une clé de chiffrement SAML

IAM utilise la clé privée que vous avez téléchargée auprès du fournisseur SAML IAM pour déchiffrer les assertions SAML chiffrées de votre IdP. Vous pouvez enregistrer jusqu'à deux fichiers de clés privées pour chaque fournisseur d'identité, ce qui vous permet d'effectuer une rotation des clés privées si nécessaire. Lorsque deux fichiers sont enregistrés, chaque requête tente d'abord d'être déchiffrée avec la date d'ajout la plus récente, puis IAM tente de déchiffrer la requête avec la date d'ajout la plus ancienne.

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation, sélectionnez Fournisseurs d'identité, puis sélectionnez votre fournisseur dans la liste.
3. Choisissez l'onglet Chiffrement SAML, puis sélectionnez Ajouter une nouvelle clé.
4. Sélectionnez Choisir un fichier et chargez la clé privée que vous avez téléchargée depuis votre IdP sous forme de fichier .pem. Ensuite, choisissez Ajouter une clé.
5. Dans la section Clés privées pour le déchiffrement SAML, sélectionnez le fichier de clé privée expiré et choisissez Supprimer. Nous vous recommandons de supprimer la clé privée expirée après avoir ajouté une nouvelle clé privée afin de garantir la réussite de la première tentative de déchiffrement de votre assertion.

Créez et gérez un fournisseur d'identité IAM SAML (AWS CLI)

Vous pouvez utiliser le AWS CLI pour créer, mettre à jour et supprimer des fournisseurs SAML. Pour obtenir de l'aide concernant la fédération SAML, consultez la section Résolution des problèmes liés à la fédération SAML.

Pour créer un fournisseur d'identité IAM et télécharger un document de métadonnées (AWS CLI)

- Exécutez cette commande : `aws iam create-saml-provider`

Pour mettre à jour un fournisseur d'identité IAM SAML (AWS CLI)

Vous pouvez mettre à jour le fichier de métadonnées, les paramètres de chiffrement SAML et effectuer la rotation des fichiers de déchiffrement par clé privée pour votre fournisseur SAML IAM. Pour faire pivoter les clés privées, ajoutez votre nouvelle clé privée, puis supprimez l'ancienne clé dans une demande distincte. Pour plus d'informations sur la rotation des clés privées, consultez Gestion des clés de chiffrement SAML.

- Exécutez cette commande : `aws iam update-saml-provider`

Pour baliser un fournisseur d'identité IAM existant (AWS CLI)

- Exécutez cette commande : `aws iam tag-saml-provider`

Pour répertorier les balises du fournisseur d'identité IAM existant (AWS CLI)

- Exécutez cette commande : `aws iam list-saml-provider-tags`

Pour supprimer les balises d'un fournisseur d'identité IAM existant (AWS CLI)

- Exécutez cette commande : `aws iam untag-saml-provider`

Pour supprimer un fournisseur d'identité IAM SAML (AWS CLI)

1. (Facultatif) Pour répertorier des informations pour tous les fournisseurs, comme l'ARN, la date de création et l'expiration, exécutez la commande suivante :

   - `aws iam list-saml-providers`

2. (Facultatif) Pour obtenir des informations sur un fournisseur spécifique, telles que l'ARN, la date de création, la date d'expiration, les paramètres de chiffrement et les informations relatives à la clé privée, exécutez la commande :

   - `aws iam get-saml-provider`

3. Pour supprimer un fournisseur d'identité IAM, exécutez la commande suivante :

   - `aws iam delete-saml-provider`

Créez et gérez un fournisseur d'identité IAM SAML (AWS API)

Vous pouvez utiliser l' AWS API pour créer, mettre à jour et supprimer des fournisseurs SAML. Pour obtenir de l'aide concernant la fédération SAML, consultez la section Résolution des problèmes liés à la fédération SAML.

Pour créer un fournisseur d'identité IAM et télécharger un document de métadonnées (AWS API)

- Appelez cette opération : `CreateSAMLProvider`

Pour mettre à jour un fournisseur d'identité IAM SAML (AWS API)

Vous pouvez mettre à jour le fichier de métadonnées, les paramètres de chiffrement SAML et effectuer la rotation des fichiers de déchiffrement par clé privée pour votre fournisseur SAML IAM. Pour faire pivoter les clés privées, ajoutez votre nouvelle clé privée, puis supprimez l'ancienne clé dans une demande distincte. Pour plus d'informations sur la rotation des clés privées, consultez Gestion des clés de chiffrement SAML.

- Appelez cette opération : `UpdateSAMLProvider`

Pour baliser un fournisseur d'identité IAM existant (AWS API)

- Appelez cette opération : `TagSAMLProvider`

Pour répertorier les balises d'un fournisseur d'identité IAM existant (AWS API)

- Appelez cette opération : `ListSAMLProviderTags`

Pour supprimer les balises d'un fournisseur d'identité IAM existant (AWS API)

- Appelez cette opération : `UntagSAMLProvider`

Pour supprimer un fournisseur d'identité IAM (AWS API)

1. (Facultatif) Pour répertorier les informations relatives à tous IdPs, telles que l'ARN, la date de création et l'expiration, effectuez l'opération suivante :
   - `ListSAMLProviders`
2. (Facultatif) Pour obtenir des informations sur un fournisseur spécifique, telles que l'ARN, la date de création, la date d'expiration, les paramètres de chiffrement et les informations relatives à la clé privée, appelez l'opération suivante :
   - `GetSAMLProvider`
3. Pour supprimer un IdP, appelez l'opération suivante :
   - `DeleteSAMLProvider`

Étapes suivantes

Après avoir créé un fournisseur d'identité SAML, configurez l'approbation des parties utilisatrices avec votre IdP. Vous pouvez également utiliser les demandes de la réponse d'authentification de votre IdP dans les politiques pour contrôler l'accès à un rôle.

- Vous devez en parler à l'IdP AWS en tant que fournisseur de services. Il s'agit là de l'ajout d'une relation d'approbation des parties utilisatrices entre votre IdP et AWS. Le processus exact utilisé pour ajouter une approbation de partie utilisatrice dépend de l'IdP que vous utilisez. Pour en savoir plus, consultez Configuration de votre IdP SAML 2.0 à l'aide d'une relation d'approbation des parties utilisatrices et ajout de demandes.
- Lorsque l'IdP envoie la réponse contenant les revendications à AWS, de nombreuses demandes entrantes sont associées à des clés de AWS contexte. Vous pouvez utiliser ces clés de contexte dans les politiques IAM à l'aide de l'élément Condition pour contrôler l'accès à un rôle. Pour plus d'informations, consultez Configuration d'assertions SAML pour la réponse d'authentification.

Configuration de votre IdP SAML 2.0 à l'aide d'une relation d'approbation des parties utilisatrices et ajout de demandes

Lorsque vous créez un fournisseur d'identité IAM et le rôle pour l'accès SAML, vous donnez des informations à AWS sur le fournisseur d'identité (IdP) externe et sur les actions que les utilisateurs sont autorisés à effectuer. La prochaine étape consiste à informer l'IdP de son statut AWS de fournisseur de services. Il s'agit là de l'ajout d'une relation d'approbation des parties utilisatrices entre votre IdP et AWS. Le processus exact utilisé pour ajouter une approbation de partie utilisatrice dépend de l'IdP que vous utilisez. Pour plus d'informations, consultez la documentation de votre logiciel de gestion des identités.

Beaucoup vous IdPs permettent de spécifier une URL à partir de laquelle l'IdP peut lire un document XML contenant des informations sur les parties utilisatrices et des certificats. Pour AWS, utilisez l'URL du point de terminaison de connexion. L'exemple suivant montre le format d'URL avec l'option `region-code`.

`https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`

Si le cryptage SAML est requis, l'URL doit inclure l'identifiant unique AWS attribué à votre fournisseur SAML, que vous trouverez sur la page détaillée du fournisseur d'identité. L'exemple suivant montre une URL de connexion régionale qui comprend un identifiant unique.

`https://region-code.signin.aws.amazon.com/static/saml/IdP-ID/saml-metadata.xml`

Pour une liste des *region-code* valeurs possibles, consultez la colonne Région dans les AWS Sign-In points de . Pour la AWS valeur, vous pouvez également utiliser le point de terminaison `https://signin.aws.amazon.com/saml` non régional.

S'il n'est pas possible de spécifier directement une URL, téléchargez le document XML à partir de l'URL ci-dessus et importez-le dans l'application de votre IdP.

Vous devez également créer des règles de réclamation appropriées dans votre IdP, qui spécifient en AWS tant que partie utilisatrice. Lorsque l'IdP envoie une réponse SAML au AWS terminal, il inclut une assertion SAML contenant une ou plusieurs revendications. Une demande contient des informations sur l'utilisateur et ses groupes. Une règle de demande mappe ces informations à des attributs SAML. Cela vous permet de vous assurer que les réponses d'authentification SAML de votre IdP contiennent les attributs nécessaires AWS utilisés dans les politiques IAM pour vérifier les autorisations des principaux fédérés SAML. Pour plus d'informations, consultez les rubriques suivantes :

- Vue d'ensemble du rôle pour autoriser SAML-federated l'accès à votre AWS resources. Cette rubrique décrit l'utilisation de SAML-specific clés dans les politiques IAM et explique comment les utiliser pour restreindre les autorisations pour les principaux fédérés SAML.
- Configuration d'assertions SAML pour la réponse d'authentification. Cette rubrique explique comment configurer des demandes SAML contenant des informations sur l'utilisateur. Les demandes sont regroupées dans une assertion SAML et intégrées dans la réponse SAML qui est envoyée à AWS. Vous devez vous assurer que les informations requises par AWS les politiques sont incluses dans l'assertion SAML sous une forme qui AWS peut être reconnue et utilisée.
- Intégrez des fournisseurs de solutions SAML tiers à AWS. Cette rubrique fournit des liens vers la documentation fournie par des organisations tierces sur la manière d'intégrer des solutions d'identité à AWS.

> **Note**
>
> Pour améliorer la résilience de la fédération, nous vous recommandons de configurer votre IdP et votre fédération AWS pour prendre en charge plusieurs points de terminaison de connexion SAML. Pour plus de détails, consultez l'article du blog sur la AWS sécurité Comment utiliser les points de terminaison SAML régionaux pour le basculement.

Intégrez des fournisseurs de solutions SAML tiers à AWS

> **Note**
>
> Nous vous recommandons de demander à vos utilisateurs humains d'utiliser des informations d'identification temporaires lors de l'accès AWS. Avez-vous envisagé de l'utiliser AWS IAM Identity Center ? Vous pouvez utiliser IAM Identity Center pour gérer de manière centralisée l'accès à plusieurs comptes Comptes AWS et fournir aux MFA-protected utilisateurs un accès en authentification unique à tous les comptes qui leur sont attribués depuis un seul endroit. Avec IAM Identity Center, vous pouvez créer et gérer les identités des utilisateurs dans IAM Identity Center ou vous connecter facilement à votre fournisseur d'identité compatible SAML 2.0 existant. Pour plus d'informations, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Les liens suivants vous aident à configurer des solutions de fournisseur d'identité (IdP) SAML 2.0 tiers pour qu'elles fonctionnent avec AWS la fédération. Vérifiez auprès de votre fournisseur d'identité s'il prend en charge le chiffrement par jeton SAML. Pour les exigences de chiffrement SAML, consultez Gestion des clés de chiffrement SAML.

> **Tip**
>
> AWS Les ingénieurs du support peuvent aider les clients qui disposent de plans de support commerciaux et d'entreprise à effectuer certaines tâches d'intégration impliquant des logiciels tiers. Pour obtenir la liste actuelle des plateformes et applications prises en charge, veuillez consulter la section What third-party software is supported? (Quels sont les logiciels tiers pris en charge ?) dans les FAQ sur AWS Support.

Solution	En savoir plus
Auth0	Intégrer à Amazon Web Services — Cette page du site Web de documentation Auth0 contient des liens vers des ressources qui décrivent comment configurer l'authent ification unique (SSO) avec le Console de gestion AWS et inclut un exemple. JavaScript Vous pouvez configurer Auth0 pour transmettre les balises de session. Pour plus d'informations, consultez Auth0 annonce un partenariat avec AWS pour les balises de session IAM.
Microsoft Entra	Tutoriel : intégration de Microsoft Entra SSO à AWS Single-Account Access — Ce didacticiel sur le site Web de Microsoft explique comment configurer Microsoft Entra (anciennement Azure AD) en tant que fournisseur d'identité (IdP) à l'aide de la fédération SAML.
Centrify	Configurer Centrify et utiliser SAML pour le SSO vers AWS — Cette page du site Web de Centrify explique comment configurer Centrify pour utiliser SAML pour le SSO à. AWS
CyberArk	Configurez CyberArk pour fournir un accès Amazon Web Services (AWS) aux utilisateurs qui se connectent via

Solution	En savoir plus
	l'authentification unique (SSO) SAML depuis le CyberArk portail utilisateur.
ForgeRock	La plateforme ForgeRock d'identité s'intègre à AWS. Vous pouvez configurer ForgeRock pour transmettre les balises de session. Pour de plus amples informations, veuillez consulter Attribute Based Access Control for Amazon Web Services.
Google Workspace	Application cloud Amazon Web Services : cet article du site d'aide destiné aux administrateurs de Google Workspace explique comment configurer Google Workspace en tant que fournisseur de services en tant que fournisseur de services en AWS tant que fournisseur de services.
IBM	Vous pouvez configurer IBM pour transmettre des balises de session. Pour plus d'informations, consultez IBM Cloud Identity IDaaS, l'un des premiers à prendre en charge les balises de AWS session.
JumpCloud	Octroi d'un accès via des rôles IAM pour l'authentification unique (SSO) avec Amazon AWS — Cet article du site JumpCloud Web explique comment configurer et activer l'authentification unique en fonction des rôles IAM pour. AWS
Matrix42	MyWorkspace Guide de démarrage — Ce guide explique comment intégrer les services d' AWS identité à MyWorkspace Matrix42.

Solution	En savoir plus
Microsoft Active Directory Federation Services (AD FS)	Notes de terrain : Intégration du service de fédération Active Directory à AWS IAM Identity Center — Cet article du blog d' AWS architecture explique le flux d'authentification entre AD FS et AWS IAM Identity Center (IAM Identity Center). IAM Identity Center prend en charge la fédération d'identité avec SAML 2.0, ce qui permet l'intégration avec les solutions AD FS. Les utilisateurs peuvent se connecter au portail IAM Identity Center avec leurs informations d'identification d'entreprise, ce qui réduit les coûts administratifs liés à la gestion des informations d'identification distinctes sur IAM Identity Center. Vous pouvez également configurer AD FS pour transmettre des balises de session. Pour de plus amples informations, veuillez consulter Use attribute-based access control with AD FS to simplify IAM permissions management.
miniOrange	SSO pour AWS — Cette page du site Web de miniOrange décrit comment établir un accès sécurisé AWS pour les entreprises et un contrôle total de AWS l'accès aux applications.
Okta	Integrating the Amazon Web Services Command Line Interface Using Okta : sur cette page du site d'assistance Okta, vous apprenez à configurer Okta en vue d'une utilisation avec AWS. Vous pouvez configurer Okta pour transmettre des balises de session. Pour plus d'informations, consultez Okta et AWS son partenariat pour simplifier l'accès via des balises de session.
Okta	AWS Fédération de comptes : cette section du site Web d'Okta explique comment configurer et activer IAM Identity Center pour. AWS

Solution	En savoir plus
OneLogin	Dans la OneLogin base de connaissances, **SAML AWS** recherchez une liste d'articles expliquant comment configurer la fonctionnalité IAM Identity Center entre OneLogin et AWS pour des scénarios à rôle unique et à rôles multiples. Vous pouvez configurer OneLogin pour transmettre les balises de session. Pour plus d'informations, consultez OneLogin la section Tags de session : Contrôle Attribute-Based d'accès aux AWS ressources.
Ping Identity	PingFederate AWS Connecteur : consultez les détails du PingFederate AWS Connector, un modèle de connexion rapide permettant de configurer facilement une connexion unique (SSO) et de provisionner une connexion. Consultez la documentation et téléchargez la dernière version du PingFederate AWS Connector pour les intégrations avec AWS. Vous pouvez configurer Ping Identity pour transmettre des balises de session. Pour plus d'informations, consultez Annonce de la prise en charge de Ping Identity pour le contrôle Attribute-Based d'accès dans AWS.
RadiantLogic	Partenaires technologiques de Radiant Logic — Le service d'identité RadiantOne fédérée de Radiant Logic s'intègre AWS pour fournir un hub d'identité pour le SAML-based SSO.
RSA	Amazon Web Services – Guide de mise en œuvre prêt pour RSA fournit des conseils pour l'intégration d' AWS et de RSA. Pour plus d'informations sur la configuration SAML, consultez Amazon Web Services – Configuration SSO de ma page SAML – Guide de mise en œuvre prêt pour RSA.

Solution	En savoir plus
Salesforce.com	Comment configurer le SSO depuis Salesforce vers AWS — Cet article pratique sur le site du Salesforce.com développeur explique comment configurer un fournisseur d'identité (IdP) dans Salesforce et le configurer en tant que fournisseur de services. AWS
SecureAuth	AWS - SSO SecureAuth SAML : cet article du site SecureAuth Web explique comment configurer l'intégration SAML avec une appliance AWS . SecureAuth
Shibboleth	Comment utiliser Shibboleth pour le SSO vers le Console de gestion AWS — Cette entrée du blog sur la AWS sécurité fournit un didacticiel étape par étape sur la façon de configurer Shibboleth et de le configurer en tant que fournisseur d'identité pour. AWS Vous pouvez configurer Shibboleth pour transmettre des balises de session.

Pour plus de détails, consultez la page des partenaires de l'IAM sur le AWS site Web.

Configuration d'assertions SAML pour la réponse d'authentification

Après avoir vérifié l'identité d'un utilisateur dans votre organisation, le fournisseur d'identité externe (IdP) envoie une réponse d'authentification à l'URL du point de terminaison de AWS connexion. La réponse est une demande POST incluant un jeton SAML conforme à la norme de liaison HTTP POST pour SAML 2.0 et contenant les éléments suivants, ou demandes. Vous configurez ces réclamations dans votre SAML-compatible IdP. Pour obtenir des instructions sur la procédure visant à entrer ces demandes, consultez la documentation de votre IdP.

Lorsque l'IdP envoie la réponse contenant les revendications à AWS, de nombreuses demandes entrantes sont associées à des clés de AWS contexte. Ces clés de contexte peuvent être vérifiées dans les politiques IAM à l'aide de l'élément `Condition`. Pour obtenir une liste des mappages disponibles, consultez la section Mappage des attributs SAML avec AWS clés de contexte de la politique de confiance.

`Objet` et `nom/ID`

La réponse doit inclure exactement 1 élément `SubjectConfirmation` avec un élément `SubjectConfirmationData` comprenant à la fois l'attribut `NotOnOrAfter` et un attribut `Recipient`. L'attribut Destinataire doit inclure une valeur qui correspond à l'URL du point de terminaison de AWS connexion. Votre IdP peut utiliser le terme `ACSRecipient`, ou `Target` faire référence à cet attribut.

Si le chiffrement SAML est obligatoire, l'URL de connexion doit inclure l'identifiant unique qu' AWS attribue à votre fournisseur SAML, que vous trouverez sur la page de détails du fournisseur d'identité. L'exemple suivant montre le format de l'URL de connexion avec l'option `region-code`

`https://`*`region-code`*`.signin.aws.amazon.com/saml`

Dans l'exemple suivant, l'URL de connexion comprend un identifiant unique, qui nécessite l'ajout de /acs/ au chemin d'accès de connexion.

`https://`*`region-code`*`.signin.aws.amazon.com/saml/acs/`*`IdP-ID`*

Pour une liste des *`region-code`* valeurs possibles, consultez la colonne Région dans les AWS Sign-In points de . Pour la AWS valeur, vous pouvez également utiliser le point de terminaison `https://signin.aws.amazon.com/saml` de connexion global.

Les éléments `NameID` peuvent avoir la valeur persistante, transitoire ou consister en l'URI de format complet tel que fourni par la solution d'IdP. La valeur persistante indique que la valeur dans `NameID` reste la même pour l'utilisateur entre les sessions. Si la valeur est transitoire, l'utilisateur a une valeur `NameID` différente pour chaque session. Les interactions avec authentification unique prennent en charge les types d'identifiants suivants :

- `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:transient`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:entity`

L'extrait suivant en présente un exemple. Substituez vos propres valeurs avec les valeurs marquées.

```
<Subject>
  <NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">_cbb88bf52c2510eabe00c1642d4643f41430fe25e3</NameID>
  <SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
    <SubjectConfirmationData NotOnOrAfter="2013-11-05T02:06:42.876Z"
 Recipient="https://region-code.signin.aws.amazon.com/saml/SAMLSP4SHN3UIS2D558H46"/>
  </SubjectConfirmation>
</Subject>
```

> **Important**
>
> La clé de contexte `saml:aud` vient de l'attribut destinataire SAML, qui est l'équivalent SAML du champ d'audience d'OIDC, par exemple `accounts.google.com:aud`.

Attribut SAML **`PrincipalTag`**

(Facultatif) Vous pouvez utiliser un élément `Attribute` dont l'attribut `Name` est défini sur `https://aws.amazon.com/SAML/Attributes/PrincipalTag:{TagKey}`. Cet élément vous permet de transmettre des attributs en tant que balises de session dans l'assertion SAML. Pour de plus amples informations sur les balises de session, veuillez consulter Transmettre les balises de session AWS STS.

Pour transmettre des attributs en tant que balises de session, incluez l'élément `AttributeValue` qui spécifie la valeur de la balise. Par exemple, pour transmettre les paires clé-valeur de balise `Project` = `Marketing` et `CostCenter` = `12345`, utilisez l'attribut suivant. Incluez un élément `Attribute` distinct pour chaque balise.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Marketing</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
```

Pour définir les balises ci-dessus comme transitives, incluez un autre élément `Attribute` dont l'attribut `Name` est défini sur `https://aws.amazon.com/SAML/Attributes/`

`TransitiveTagKeys`. Il s'agit d'un attribut facultatif à valeurs multiples qui définit vos balises de session comme transitives. Les balises transitives persistent lorsque vous utilisez la session SAML pour endosser un autre rôle dans AWS. Ceci est connu sous le nom de chaînage de rôles. Par exemple, pour définir les balises `Principal` et `CostCenter` comme transitives, utilisez l'attribut suivant pour spécifier les clés.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>CostCenter</AttributeValue>
</Attribute>
```

`Attribut` SAML du rôle

Vous pouvez utiliser un élément `Attribute` avec l'attribut `Name` défini sur `https://aws.amazon.com/SAML/Attributes/Role` Cet élément contient un ou plusieurs éléments `AttributeValue` qui répertorient le fournisseur d'identité et le rôle IAM auxquels l'utilisateur est mappé par votre IdP. Le rôle IAM et le fournisseur d'identité IAM sont spécifiés sous la forme d'une paire d'ARN séparés par des virgules dans le même format que les `RoleArn` paramètres et qui sont transmis. `PrincipalArn` AssumeRoleWithSAML Cet élément doit contenir au moins une paire de fournisseur de rôles (élément `AttributeValue`) et peut contenir plusieurs paires. Si l'élément contient plusieurs paires, l'utilisateur doit choisir le rôle à endosser lorsqu'il utilise WebSSO afin de se connecter à la Console de gestion AWS.

> **Important**
>
> La valeur de l'attribut `Name` dans la balise `Attribute` est sensible à la casse. Il doit être défini sur `https://aws.amazon.com/SAML/Attributes/Role` précisément.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/Role">
  <AttributeValue>arn:aws:iam::account-number:role/role-name1,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name2,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name3,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
</Attribute>
```

Attribut SAML **RoleSessionName**

Vous pouvez utiliser un élément `Attribute` avec l'attribut `Name` défini sur `https://aws.amazon.com/SAML/Attributes/RoleSessionName` Cet élément contient un élément `AttributeValue` qui fournit un identifiant pour les informations d'identification temporaires qui sont émises lorsque le rôle est endossé. Vous pouvez l'utiliser pour associer les informations d'identification temporaires à l'utilisateur qui utilise votre application. Cet élément est utilisé pour afficher les informations utilisateur dans le Console de gestion AWS. La valeur de l'élément `AttributeValue` doit comporter entre 2 et 64 caractères. Elle ne peut contenir que des caractères alphanumériques, des traits de soulignement et les caractères suivants : . , + = @ - (tiret). Il ne doit pas contenir d'espace. La valeur est généralement un ID utilisateur (`john`) ou une adresse e-mail (`johndoe@example.com`). Il ne peut pas contenir d'espace, comme dans le nom complet de l'utilisateur (`John Doe`).

> **Important**
>
> La valeur de l'attribut `Name` dans la balise `Attribute` est sensible à la casse. Il doit être défini sur `https://aws.amazon.com/SAML/Attributes/RoleSessionName` précisément.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/RoleSessionName">
  <AttributeValue>user-id-name</AttributeValue>
</Attribute>
```

Attribut SAML **SessionDuration**

(Facultatif) Vous pouvez utiliser un élément `Attribute` dont l'attribut `Name` est défini sur `https://aws.amazon.com/SAML/Attributes/SessionDuration`. Cet élément contient un `AttributeValue` élément qui spécifie la durée pendant laquelle l'utilisateur peut accéder Console de gestion AWS avant de devoir demander de nouvelles informations d'identification temporaires. La valeur est un nombre entier représentant le nombre de secondes pour la session. La valeur peut être comprise entre 900 secondes (15 minutes) et 43 200 secondes (12 heures). Si cet attribut n'est pas présent, la durée des informations d'identification est d'une d'heure (valeur par défaut du paramètre `DurationSeconds` de l'API `AssumeRoleWithSAML`).

Pour utiliser cet attribut, vous devez configurer le fournisseur SAML pour fournir un accès Console de gestion AWS par authentification unique au point de terminaison Web de connexion à la console situé

à. `https://region-code.signin.aws.amazon.com/saml` Pour une liste des *region-code* valeurs possibles, consultez la colonne Région dans les AWS Sign-In points de . Vous pouvez éventuellement utiliser l'URL suivante :`https://signin.aws.amazon.com/static/saml`. Notez que cet attribut prolonge les sessions uniquement à la Console de gestion AWS. Il ne peut pas prolonger la durée de vie des autres informations d'identification. Cependant, s'il est présent dans un appel d'API `AssumeRoleWithSAML`, il peut être utilisé pour raccourcir la durée de la session. La durée de vie par défaut des informations d'identification renvoyées par l'appel est de 60 minutes.

Notez également que si un attribut `SessionNotOnOrAfter` est également défini, la valeur la plus faible des deux attributs `SessionDuration` ou `SessionNotOnOrAfter` établit la durée maximale de la session de console.

Lorsque vous activez les sessions de console avec une durée prolongée, le risque de divulgation des informations d'identification augmente. Pour vous aider à réduire ce risque, vous pouvez immédiatement désactiver les sessions de console actives pour un rôle en sélectionnant Revoke Sessions (Révoquer les sessions) sur la page Role Summary (Résumé du rôle) de la console IAM. Pour plus d'informations, veuillez consulter Révocation des informations d'identification de sécurité temporaires d'un rôle IAM.

> **Important**
>
> La valeur de l'attribut `Name` dans la balise `Attribute` est sensible à la casse. Il doit être défini sur `https://aws.amazon.com/SAML/Attributes/SessionDuration` précisément.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SessionDuration">
  <AttributeValue>1800</AttributeValue>
</Attribute>
```

Attribut SAML **`SourceIdentity`**

(Facultatif) Vous pouvez utiliser un élément `Attribute` dont l'attribut `Name` est défini sur `https://aws.amazon.com/SAML/Attributes/SourceIdentity`. Cet élément contient un élément `AttributeValue` qui fournit un identifiant pour la personne ou l'application qui utilise un rôle IAM. La valeur de l'identité de source persiste lorsque vous utilisez la session SAML pour assumer un autre rôle dans ce que l' AWS on appelle le chaînage de rôles. La valeur de l'identité source est présente dans la demande pour chaque action effectuée durant la session de rôle. La valeur définie

ne peut pas être modifiée durant la session de rôle. Les administrateurs peuvent ensuite utiliser AWS CloudTrail les journaux pour surveiller et auditer les informations d'identité de la source afin de déterminer qui a effectué des actions avec des rôles partagés.

La valeur de l'`AttributeValue`élément doit comporter entre 2 et 256 caractères et ne peut contenir que des caractères alphanumériques, des traits de soulignement et les caractères suivants :, + = @ - (tiret). Il ne doit pas contenir d'espace. La valeur est généralement un attribut associé à l'utilisateur, comme un id utilisateur (`john`) ou une adresse e-mail (`johndoe@example.com`). Il ne peut pas contenir d'espace, comme dans le nom complet de l'utilisateur (`John Doe`). Pour de plus amples informations sur l'utilisation de l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.

> **Important**
>
> Si votre assertion SAML est configurée pour utiliser l'attribut `SourceIdentity`, votre politique d'approbation de rôle doit également inclure l'action `sts:SetSourceIdentity`, sinon l'opération endosser le rôle échouera. Pour de plus amples informations sur l'utilisation de l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.

Pour transmettre un attribut d'identité source, incluez l'élément `AttributeValue` qui spécifie la valeur de l'identité source. Par exemple, pour transmettre l'identité source `Diego`, utilisez l'attribut suivant.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
  <AttributeValue>Diego</AttributeValue>
```

Mappage des attributs SAML avec AWS clés de contexte de la politique de confiance

Les tableaux de cette section répertorient les attributs SAML utilisés couramment et leur correspondance avec les clés de contexte de condition de politique d'approbation dans AWS. Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés aux valeurs incluses dans les assertions qui accompagnent une demande d'accès SAML.

> **Important**
>
> Ces clés sont disponibles uniquement dans les politiques d'approbation IAM (politiques qui déterminent qui peut endosser un rôle) et ne sont pas applicables aux politiques d'autorisations.

Dans le tableau des attributs eduPerson et eduOrg, les valeurs sont saisies sous forme de chaînes ou de listes de chaînes. Pour les valeurs de chaînes, vous pouvez tester ces valeurs dans des politiques d'approbation IAM à l'aide des conditions `StringEquals` ou `StringLike`. Concernant les valeurs qui contiennent une liste de chaînes, vous pouvez utiliser les opérateurs d'ensemble de politique `ForAnyValue` et `ForAllValues` pour tester les valeurs dans les politiques de confiance.

> **Note**
>
> Vous ne devez inclure qu'une seule revendication par clé de AWS contexte. Si vous incluez plusieurs demandes, une seule d'entre elles sera mappée.

Le tableau suivant présente les attributs eduPerson et eduOrg.

Attribut eduPerson ou eduOrg (**`Name`** clé)	Correspond à cette clé AWS contextuelle (**`FriendlyName`** clé)	Type
`urn:oid:1.3.6.1.4.1.5923.1.1.1.1`	`eduPerson Affiliation`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.1.1.2`	`eduPersonNickname`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.1.1.3`	`eduPersonOrgDN`	Chaîne
`urn:oid:1.3.6.1.4.1.5923.1.1.1.4`	`eduPerson OrgUnitDN`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.1.1.5`	`eduPerson PrimaryAf filiation`	Chaîne

Attribut eduPerson ou eduOrg (**`Name`** clé)	Correspond à cette clé AWS contextuelle (**`FriendlyName`** clé)	Type
`urn:oid:1.3.6.1.4.1.5923.1.1.1.6`	`eduPerson PrincipalName`	Chaîne
`urn:oid:1.3.6.1.4.1.5923.1.1.1.7`	`eduPerson Entitlement`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.1.1.8`	`eduPerson PrimaryOrgUnitDN`	Chaîne
`urn:oid:1.3.6.1.4.1.5923.1.1.1.9`	`eduPerson ScopedAff iliation`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1. 1.1.10`	`eduPerson TargetedID`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1. 1.1.11`	`eduPerson Assurance`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.2.1.2`	`eduOrgHomePageURI`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.2.1.3`	`eduOrgIde ntityAuth NPolicyURI`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.2.1.4`	`eduOrgLegalName`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.2.1.5`	`eduOrgSuperiorURI`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.2.1.6`	`eduOrgWhi tePagesURI`	Liste de chaînes
`urn:oid:2.5.4.3`	`cn`	Liste de chaînes

Le tableau suivant présente les attributs Active Directory.

Attribut AD	Correspond à cette clé AWS contextuelle	Type
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name`	`name`	Chaîne
`http://schemas.xmlsoap.org/claims/CommonName`	`commonName`	Chaîne
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname`	`givenName`	Chaîne
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname`	`surname`	Chaîne
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`	`mail`	Chaîne
`http://schemas.microsoft.com/ws/2008/06/identity/claims/primarygroupsid`	`uid`	Chaîne

Le tableau suivant présente X.500 les attributs.

X.500 attribut	Correspond à cette clé AWS contextuelle	Type
`2.5.4.3`	`commonName`	Chaîne
`2.5.4.4`	`surname`	Chaîne
`2.4.5.42`	`givenName`	Chaîne
`2.5.4.45`	`x500UniqueIdentifier`	Chaîne
`0.9.2342.19200300100.1.1`	`uid`	Chaîne

X.500 attribut	Correspond à cette clé AWS contextuelle	Type
`0.9.2342.19200300100.1.3`	`mail`	Chaîne
`0.9.2342.19200300.100.1.45`	`organizationStatus`	Chaîne

Permettre aux principaux fédérés SAML 2.0 d'accéder au Console de gestion AWS

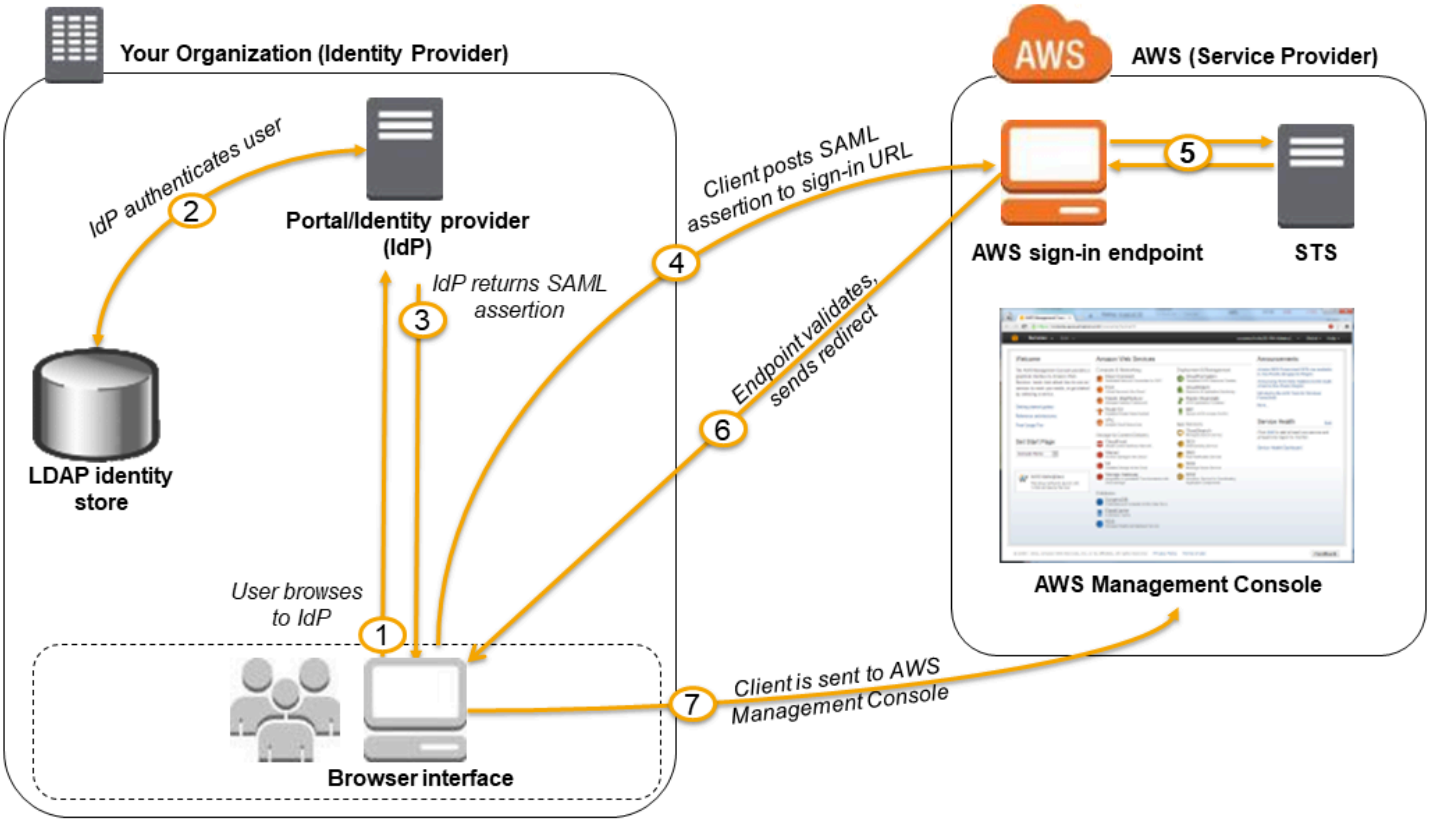
Vous pouvez utiliser un rôle pour configurer votre fournisseur d'identité (IdP) conforme à SAML 2.0 et AWS pour autoriser les principaux fédérés SAML à accéder au. Console de gestion AWS Le rôle accorde aux utilisateurs les autorisations nécessaires pour effectuer des tâches dans la console. Si vous souhaitez donner aux principaux fédérés SAML d'autres moyens d'accès AWS, consultez l'une des rubriques suivantes :

- AWS CLI: Passez à un rôle IAM (AWS CLI)
- Outils pour Windows PowerShell : Passer à un rôle IAM (Outils pour Windows PowerShell)
- AWS API : Passez à un rôle IAM (AWS API)

Présentation de

Le schéma suivant illustre le flux de l'authentification SAML-enabled unique.

> **Note**
>
> Cette utilisation spécifique de SAML diffère de l'utilisation plus générale illustrée ici Fédération SAML 2.0 car ce flux de travail ouvre le fichier Console de gestion AWS au nom de l'utilisateur. Cette procédure requiert l'utilisation du point de terminaison SSO AWS au lieu d'un appel direct de l'API `AssumeRoleWithSAML`. Le point de terminaison appelle l'API pour l'utilisateur, puis retourne une URL qui redirige automatiquement le navigateur de l'utilisateur vers Console de gestion AWS.



Le diagramme suivant illustre les étapes suivantes :

1. L'utilisateur se rend sur le portail de l'organisation, puis choisit d'accéder à l'interface Console de gestion AWS. Dans votre organisation, le portail est généralement une fonction de votre IdP qui gère l'échange de confiance entre votre organisation et. AWS Par exemple, dans les services Active Directory Federation Services, l'URL du portail est : `https://ADFSServiceName/adfs/ls/IdpInitiatedSignOn.aspx`
2. Le portail vérifie l'identité de l'utilisateur dans l'organisation.
3. Il génère ensuite une réponse d'authentification SAML contenant des assertions qui identifient l'utilisateur et incluent des attributs spécifiques à celui-ci. Vous pouvez également configurer votre IdP pour inclure un attribut d'assertion SAML appelé `SessionDuration` qui spécifie la durée de validité de la session de console. Vous pouvez également configurer le fournisseur d'identité pour qu'il transmette des attributs en tant que balises de session. Le portail envoie cette réponse au navigateur client.
4. Le navigateur client est redirigé vers le point de terminaison d'authentification AWS unique et publie l'assertion SAML.
5. Le point de terminaison demande des informations d'identification de sécurité temporaires pour le compte de l'utilisateur et crée une URL de connexion à la console qui utilise ces informations.

6. AWS renvoie l'URL de connexion au client sous forme de redirection.
7. Le navigateur client est redirigé vers Console de gestion AWS. Si la réponse d'authentification SAML inclut des attributs associés à plusieurs rôles IAM, l'utilisateur est d'abord invité à sélectionner le rôle utilisé pour accéder à la console.

Du point de vue de l'utilisateur, le processus se déroule de manière transparente : l'utilisateur commence sur le portail interne de votre organisation et se retrouve sur le portail Console de gestion AWS, sans jamais avoir à fournir d'informations d' AWS identification.

Consultez les sections suivantes pour une présentation de la configuration de ce comportement, ainsi que des liens vers des procédures détaillées.

Configurez votre réseau en tant que fournisseur SAML pour AWS

Au sein du réseau de votre organisation, vous configurez votre magasin d'identités (tel que Windows Active Directory) pour qu'il fonctionne avec un SAML-based IdP tel que Windows Active Directory Federation Services, Shibboleth, etc. À l'aide de l'IdP, vous générez un document de métadonnées qui décrit votre organisation en tant que fournisseur d'identité et inclut des clés d'authentification. Vous configurez également le portail de votre organisation pour acheminer les demandes des utilisateurs vers le point de terminaison AWS SAML Console de gestion AWS à des fins d'authentification à l'aide d'assertions SAML. La configuration de l'IdP pour la génération d'un fichier metadata.xml varie en fonction de l'IdP. Pour savoir comment procéder, reportez-vous à la documentation de votre IdP, ou consultez Intégrez des fournisseurs de solutions SAML tiers à AWS pour obtenir des liens vers la documentation web de nombreux fournisseurs SAML pris en charge.

Créer un fournisseur SAML dans IAM

Ensuite, vous vous connectez à la console IAM Console de gestion AWS et vous y accédez. Dans la console, vous créez un nouveau fournisseur SAML, c'est-à-dire une entité IAM qui contient des informations sur le fournisseur d'identité de votre organisation. Au cours du processus, vous téléchargez le document de métadonnées généré par le logiciel de l'IdP de votre organisation à l'étape précédente. Pour en savoir plus, consultez Création d'un fournisseur d'identité SAML dans IAM.

Configurer les autorisations dans AWS pour les principes fédérés SAML

L'étape suivante consiste à créer un rôle IAM qui établit une relation d'approbation entre IAM et le fournisseur d'identités de votre organisation. Ce rôle doit identifier votre fournisseur d'identité en tant

que principal (entité de confiance) pour les besoins de la fédération. Le rôle définit également ce que les utilisateurs authentifiés par l'IdP de votre organisation sont autorisés à faire. AWS Vous pouvez créer ce rôle à l'aide de la console IAM. Lorsque vous créez la politique d'approbation qui indique qui peut endosser le rôle, vous indiquez le fournisseur SAML que vous avez créé précédemment dans IAM. Vous indiquez également un ou plusieurs attributs SAML auxquels un utilisateur doit correspondre pour être autorisé à endosser le rôle. Par exemple, vous pouvez spécifier que seuls les utilisateurs dont la valeur `eduPersonOrgDN` SAML est `ExampleOrg` sont autorisés à se connecter. L'assistant de rôle ajoute automatiquement une condition pour tester l'attribut `saml:aud` afin de vérifier que le rôle est endossé uniquement pour la connexion à Console de gestion AWS.

Si le chiffrement SAML est obligatoire, l'URL de connexion doit inclure l'identifiant unique qu' AWS attribue à votre fournisseur SAML, que vous trouverez sur la page de détails du fournisseur d'identité. La politique d'approbation du rôle peut se présenter comme suit :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:saml-provider/
ExampleOrgSSOProvider"
            },
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "saml:edupersonorgdn": "ExampleOrg",
                    "saml:aud": "https://region-code.signin.aws.amazon.com/saml/
acs/SAMLSP4SHN3UIS2D558H46"
                }
            }
        }
    ]
}
```

> **Note**
>
> Les IdP SAML utilisés dans une politique d'approbation de rôle doivent se trouver dans le même compte que le rôle.

Nous recommandons d'utiliser des points de terminaison régionaux pour l'attribut `saml:aud` ici `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`. Pour obtenir la liste des *region-code* valeurs possibles, consultez la colonne Région dans les AWS Sign-In points de .

Dans la politique d'autorisation du rôle, vous spécifiez les autorisations comme pour tout autre rôle, utilisateur ou groupe. Par exemple, si les utilisateurs de votre organisation sont autorisés à administrer les instances Amazon EC2, vous autorisez explicitement les actions Amazon EC2 dans la politique d'autorisation. Pour ce faire, vous pouvez affecter une politique gérée comme Amazon EC2 Full Access.

Pour plus d'informations sur la création d'un rôle pour un IdP SAML, consultez Création d'un rôle pour la fédération SAML 2.0 (console).

Terminer la configuration et créer des assertions SAML

Indiquez à votre IdP SAML AWS qu'il s'agit de votre fournisseur de services en installant `saml-metadata.xml` le fichier trouvé `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml` sur ou. `https://signin.aws.amazon.com/static/saml-metadata.xml` Si le chiffrement SAML est requis, le fichier se trouve à `https://region-code.signin.aws.amazon.com/static/saml/SAMLSP4SHN3UIS2D558H46/saml-metadata.xml` l'adresse.

Pour obtenir la liste des *region-code* valeurs possibles, consultez la colonne Région dans les AWS Sign-In points de .

L'installation du fichier est différente selon votre IdP. Certains fournisseurs vous permettent d'entrer l'URL, après quoi ils récupèrent et installent le fichier pour vous. D'autres fournisseurs exigent que vous téléchargiez le fichier à partir de l'URL afin de le fournir en tant que fichier local. Pour plus d'informations, reportez-vous à la documentation de votre IdP, ou consultez Intégrez des fournisseurs de solutions SAML tiers à AWS pour obtenir des liens vers la documentation web de nombreux fournisseurs SAML pris en charge.

Vous devez également configurer les informations que l'IdP doit transmettre à AWS en tant qu'attributs SAML dans le cadre de la réponse d'authentification. La plupart de ces informations apparaissent AWS sous forme de clés contextuelles de condition que vous pouvez évaluer dans vos politiques. Ces clés de condition garantissent que seuls les utilisateurs autorisés dans les contextes appropriés sont autorisés à accéder à vos ressources AWS . Vous pouvez spécifier des fenêtres horaires qui limitent l'utilisation de la console. Vous pouvez également spécifier la durée maximale (jusqu'à 12 heures) pendant laquelle les utilisateurs peuvent accéder à la console avant d'avoir à actualiser leurs informations d'identification. Pour en savoir plus, consultez Configuration d'assertions SAML pour la réponse d'authentification.

Affichage d'une réponse SAML dans votre navigateur

Les procédures suivantes décrivent l'affichage de la réponse SAML de votre fournisseur de services dans votre navigateur lors du dépannage d'un problème lié à SAML 2.0.

Pour tous les navigateurs, accédez à la page où vous pouvez reproduire le problème. Ensuite, suivez les étapes correspondant au navigateur approprié :

Rubriques

- Google Chrome
- Mozilla Firefox
- Apple Safari
- Que faire avec la réponse Base64-encoded SAML

Google Chrome

Pour afficher une réponse SAML dans Chrome

Ces étapes ont été testées à l'aide de la version 106.0.5249.103 (build officielle) (arm64) de Google Chrome. Si vous utilisez une autre version, il vous faudra peut-être modifier les étapes en conséquence.

1. Appuyez sur F12 pour démarrer la console Outils pour développeurs.
2. Sélectionnez l'onglet Réseau, puis sélectionnez Conserver le journal en haut à gauche de la fenêtre Outils pour développeurs.
3. Reproduisez le problème.

4. (Facultatif) Si la colonne Méthode n'est pas visible dans le volet journal du réseau Developer Tools, ouvrez le menu contextuel (clic droit) correspondant à n'importe quel libellé de colonne et choisissez Méthode pour ajouter la colonne.
5. Recherchez une Publication SAML dans le volet du journal Outils pour développeurs Réseau. Sélectionnez cette ligne, puis affichez l'onglet Charge utile en haut. Recherchez l'élément SAMLResponse contenant la demande encodée. La valeur associée est la Base64-encoded réponse.

Mozilla Firefox

Pour afficher une réponse SAML dans Firefox

Cette procédure a été testée avec la version 105.0.3 (64 bits) de Mozilla Firefox. Si vous utilisez une autre version, il vous faudra peut-être modifier les étapes en conséquence.

1. Appuyez sur F12 pour démarrer la console Outils Web Developer.
2. Sélectionnez l'onglet Réseau.
3. Dans le coin supérieur droit de la fenêtre Outils Web Developer, cliquez sur options (petite icône d'engrenage). Sélectionnez Conserver les journaux.
4. Reproduisez le problème.
5. (Facultatif) Si la colonne Méthode n'est pas visible dans le volet journal Web Developer Tools Network, ouvrez le menu contextuel (clic droit) correspondant à n'importe quel libellé de colonne et choisissez Méthode pour ajouter la colonne.
6. Recherchez un POST SAML dans le tableau. Sélectionnez cette ligne, puis affichez l'onglet Requête et trouvez l'élément SamlResponse. La valeur associée est la Base64-encoded réponse.

Apple Safari

Pour afficher une réponse SAML dans Safari

Ces étapes ont été testées à l'aide de la version 16.0 (17614.1.25.9.10, 17614) d'Apple Safari. Si vous utilisez une autre version, il vous faudra peut-être modifier les étapes en conséquence.

1. Activez Web Inspector dans Safari. Ouvrez la fenêtre Préférences, sélectionnez l'onglet Avancées, puis sélectionnez Afficher le menu Développement dans la barre de menus.

2. Vous pouvez maintenant ouvrir Web Inspector. Choisissez Développement dans la barre de menu, puis sélectionnez Afficher Web Inspector.
3. Sélectionnez l'onglet Réseau.
4. Dans l'angle supérieur gauche de la fenêtre Web Inspector, choisissez les options (l'icône représentant un petit cercle contenant trois lignes horizontales). Sélectionnez Conserver le journal.
5. (Facultatif) Si la colonne Méthode n'est pas visible dans le volet journal Web Inspector Network, ouvrez le menu contextuel (clic droit) correspondant à n'importe quel libellé de colonne et choisissez Méthode pour ajouter la colonne.
6. Reproduisez le problème.
7. Recherchez un POST SAML dans le tableau. Sélectionnez cette ligne, puis affichez l'onglet En-têtes.
8. Recherchez l'élément SAMLResponse contenant la demande encodée. Faites défiler la liste pour localiser l'élément `Request Data` nommé `SAMLResponse`. La valeur associée est la Base64-encoded réponse.

Que faire avec la réponse Base64-encoded SAML

Une fois que vous avez trouvé l'élément de réponse Base64-encoded SAML dans votre navigateur, copiez-le et utilisez votre outil de Base-64 décodage préféré pour extraire la réponse balisée XML.

> **Conseil de sécurité**
>
> Dans la mesure où les données de réponse SAML que vous affichez peuvent être des données de sécurité sensibles, il est recommandé de ne pas utiliser un décodeur Base64 en ligne. Préférez un outil installé sur un ordinateur local qui n'achemine pas les données SAML via le réseau.

Built-in option pour les systèmes Windows (PowerShell) :

```
PS C:
\> [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String("base64encodedtext
```

Built-in option pour les systèmes macOS et Linux :

```
$ echo "base64encodedtext" | base64 --decode
```

Révision des valeurs dans le fichier décodé

Révisez les valeurs dans le fichier de réponse SAML décodé

- Vérifiez que la valeur de l'attribut saml:NameID correspond au nom d'utilisateur de l'utilisateur authentifié.
- Vérifiez la valeur de https://aws.amazon.com/SAML/Attributes/Role. L'ARN et du fournisseur SAML distinguent les majuscules et minuscules, et l'ARN doit correspondre à la ressource de votre compte.
- Vérifiez la valeur de https://aws.amazon.com/SAML/Attributes/RoleSessionName. La valeur doit correspondre à la valeur indiquée dans la règle de réclamation.
- Si vous configurez la valeur d'attribut pour une adresse e-mail ou un nom de compte, assurez-vous que les valeurs sont correctes. Les valeurs doivent correspondre à l'adresse e-mail ou au nom de compte de l'utilisateur authentifié.

Cérification des erreurs et confirmation de la configuration

Vérifiez si les valeurs contiennent des erreurs et confirmez que les configurations suivantes sont correctes.

- Les règles de demande répondent aux éléments requis et tous les ARN sont corrects. Pour de plus amples informations, veuillez consulter Configuration de votre IdP SAML 2.0 à l'aide d'une relation d'approbation des parties utilisatrices et ajout de demandes.
- Vous avez chargé le dernier fichier de métadonnées de votre IdP vers votre AWS fournisseur SAML. Pour de plus amples informations, veuillez consulter Permettre aux principaux fédérés SAML 2.0 d'accéder au Console de gestion AWS.
- Vous avez correctement configuré la politique de confiance du rôle IAM. Pour de plus amples informations, veuillez consulter Méthodes pour assumer un rôle.

Fédérer AWS Identités vis-à-vis des services externes

La fédération d'identités sortantes IAM permet à vos AWS charges de travail d'accéder en toute sécurité à des services externes sans stocker d'informations d'identification à long terme. Vos AWS charges de travail peuvent demander des jetons Web JSON (JWT) de courte durée auprès de AWS

Security Token Service (AWS STS) en appelant l'API. GetWebIdentityToken Ces jetons sont signés cryptographiquement, vérifiables publiquement et contiennent un ensemble complet de déclarations qui affirment l'identité de votre AWS charge de travail auprès de services externes. Vous pouvez utiliser ces jetons avec un large éventail de fournisseurs de cloud tiers, de plateformes SaaS et d'applications auto-hébergées. Les services externes vérifient l'authenticité du jeton à l'aide AWS de clés de vérification publiées sur des terminaux connus et utilisent les informations contenues dans les jetons pour prendre des décisions d'authentification et d'autorisation.

La fédération des identités sortantes élimine le besoin de stocker des informations d'identification à long terme telles que des clés d'API ou des mots de passe dans le code de votre application ou dans des variables d'environnement, améliorant ainsi votre niveau de sécurité. Vous pouvez contrôler l'accès à la génération de jetons et appliquer les propriétés des jetons telles que les algorithmes de signature, les audiences autorisées et la durée à l'aide de politiques IAM. Toutes les demandes de jetons sont enregistrées AWS , fournissant des pistes d'audit complètes pour la surveillance de la sécurité et les rapports de conformité. Vous pouvez également personnaliser les jetons avec des balises qui apparaissent sous forme de revendications personnalisées, ce qui permet aux services externes de mettre en œuvre un contrôle d'accès précis basé sur les attributs.

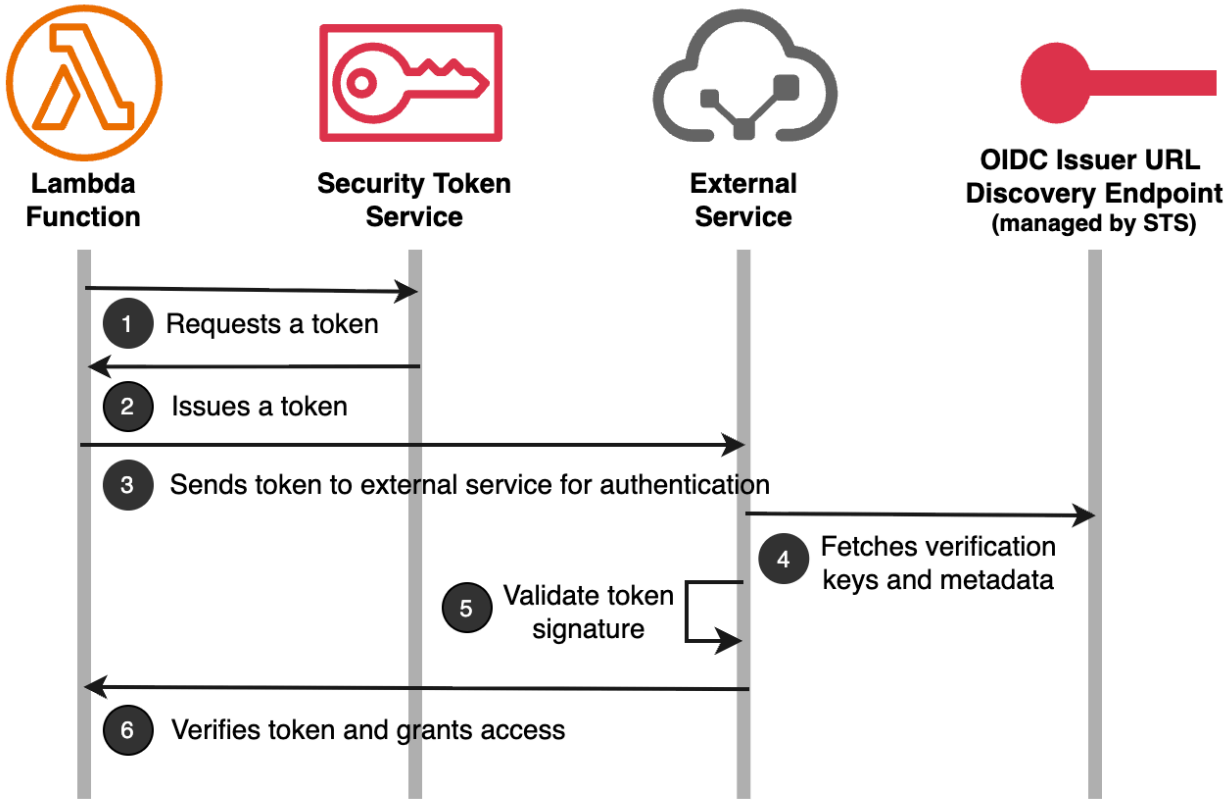
Pour obtenir des instructions sur la configuration de la fédération sortante, consultezCommencer à utiliser la fédération des identités sortantes. Pour plus d'informations sur le contenu des jetons, consultezComprendre les revendications relatives aux jetons. Pour obtenir des instructions sur le contrôle des charges de travail qui peuvent demander des jetons, consultezContrôler l'accès à l'aide de politiques IAM.

Cas d'utilisation courants

Grâce à la fédération des identités sortantes, vos AWS charges de travail peuvent en toute sécurité :

- Accédez à des ressources et à des services auprès de fournisseurs de cloud externes. Par exemple, une fonction Lambda traitant des données peut écrire les résultats dans le service de stockage d'un fournisseur de cloud externe ou interroger sa base de données.
- Intégrez des fournisseurs externes de logiciels en tant que service (SaaS) pour l'analyse, le traitement des données, la surveillance, etc. Par exemple, vos fonctions Lambda peuvent envoyer des métriques à des plateformes d'observabilité.
- Authentifiez-vous avec vos propres applications hébergées sur AWS des fournisseurs de cloud externes ou des centres de données sur site, permettant ainsi des architectures hybrides et multicloud sécurisées. Par exemple, vos AWS charges de travail peuvent interagir avec des applications conteneurisées exécutées dans votre cluster Kubernetes sur site.

Comment ça marche



1. La fonction Lambda appelle l'GetWebIdentityTokenAPI pour demander un jeton Web JSON (JWT) auprès de AWS Security Token Service (AWS STS).
2. AWS STS valide la demande et renvoie un JWT signé à la fonction Lambda.
3. La fonction Lambda envoie le JWT au service externe.
4. Le service externe extrait l'URL de l'émetteur du jeton, vérifie qu'elle correspond à un émetteur fiable connu et récupère les clés de vérification et les métadonnées depuis le point AWS de terminaison de découverte OIDC.
5. Le service externe utilise les clés de vérification pour vérifier la signature du jeton et valide les déclarations telles que l'heure d'expiration, le sujet et le public.
6. Une fois la validation réussie, le service externe autorise l'accès à la fonction Lambda.

Commencer à utiliser la fédération des identités sortantes

Ce guide explique comment activer la fédération d'identité sortante pour votre AWS compte et obtenir votre premier jeton Web JSON (JWT) (à l'aide de l'GetWebIdentityTokenAPI). Vous allez activer la

fonctionnalité, établir une relation de confiance avec un service externe, configurer les autorisations IAM et demander un jeton à l'aide de la AWS CLI ou du AWS SDK pour Python (Boto3).

Conditions préalables

Avant de commencer, assurez-vous d'avoir :

- Dernière version de la AWS CLI ou Python 3.8 (ou version ultérieure) et Boto3 installés (pour les exemples de AWS SDK)
- Un compte de service externe dans lequel vous pouvez configurer des relations de confiance (comme un fournisseur de cloud externe, un fournisseur SaaS ou une application de test)

> **Note**
>
> - L'`GetWebIdentityToken`API n'est pas disponible sur le terminal STS Global.
> - Les jetons Web JSON (JWT) générés par l'`GetWebIdentityToken`API ne peuvent pas être utilisés pour la fédération OpenID Connect (OIDC) vers AWS (via l'API). AssumeRoleWithWebIdentity

Activez la fédération des identités sortantes pour votre compte

Vous devez activer la fédération des identités sortantes avant de pouvoir demander des jetons. Vous pouvez activer la fonctionnalité à l'aide de la console AWS de gestion ou par programmation à l'aide de l'EnableOutboundWebIdentityFederationAPI.

Utilisation de AWS INTERFACE DE LIGNE DE COMMANDE (CLI)

```
aws iam enable-outbound-web-identity-federation
```

Utilisation AWS SDK pour Python

```
import boto3

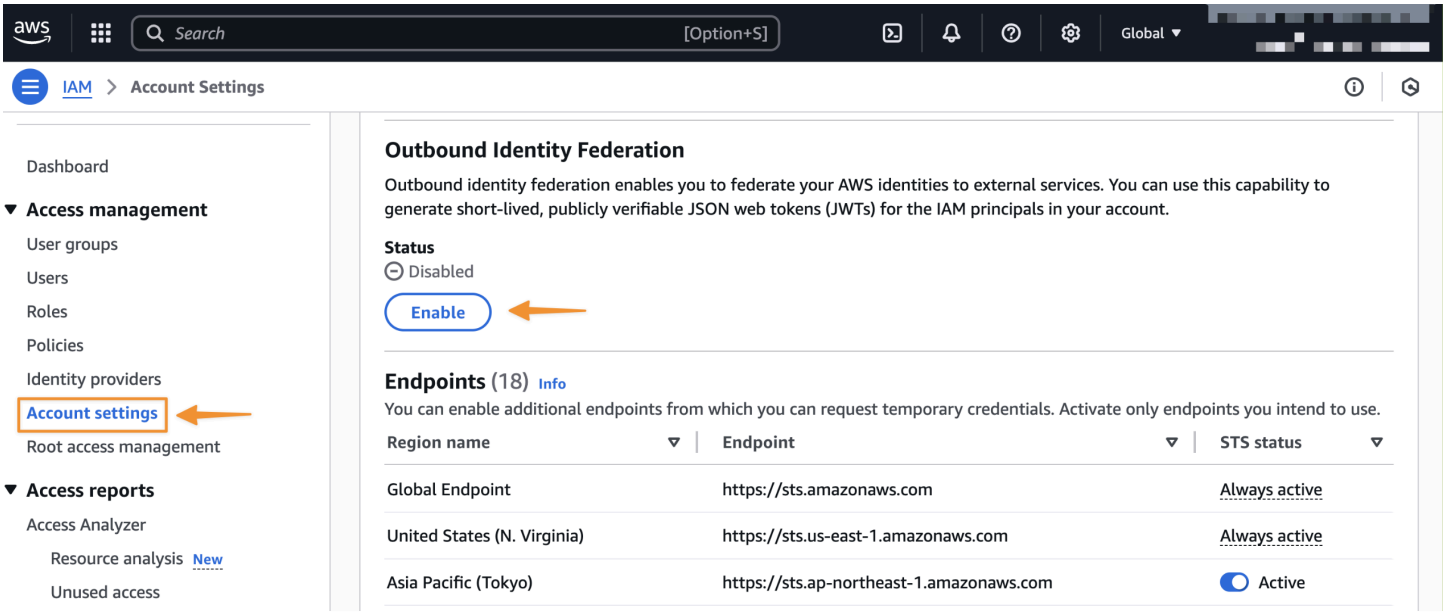
# Create IAM client
iam_client = boto3.client('iam')

# Enable outbound identity federation
response = iam_client.enable_outbound_web_identity_federation()
```

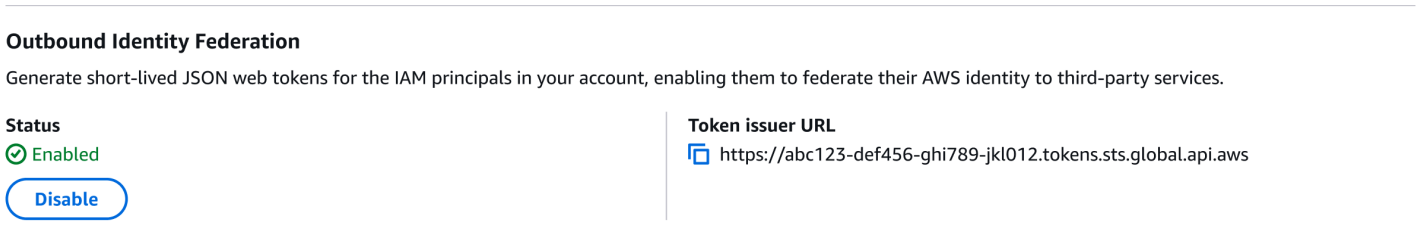
```
print(f"Feature enabled. Issuer URL: {response['IssuerUrl']}")
print(f"Status: {response['Status']}")
```

Utilisation de AWS Console

Accédez à IAM et sélectionnez Paramètres du compte dans la section Gestion des accès du menu de navigation de gauche



Une fois la fonctionnalité activée, notez l'URL de l'émetteur spécifique à votre compte. Vous utiliserez cette URL lors de la configuration des relations de confiance dans les services externes. Vous pouvez également récupérer l'URL de cet émetteur selon vos besoins à l'aide de l'GetOutboundWebIdentityFederationInfoAPI.



Etablir une relation de confiance dans le service externe

Configurez le service externe pour qu'il approuve et accepte les jetons émis par votre AWS compte. Les étapes spécifiques varient selon le service, mais impliquent généralement :

- Enregistrement de l'URL de l'émetteur de votre AWS compte en tant que fournisseur d'identité de confiance

- Configuration des allégations à valider (public, modèles de sujets)
- Associer les revendications de jetons aux autorisations du service externe

Reportez-vous à la documentation du service externe pour obtenir des instructions de configuration détaillées.

Configuration des autorisations IAM

Créez une politique IAM qui autorise l'appel à l'GetWebIdentityTokenAPI et associez la politique à un rôle IAM qui doit générer des jetons.

Cet exemple de politique accorde l'accès à la génération de jetons avec des restrictions spécifiques. Il permet de demander des jetons uniquement pour « https://api.example.com » en tant qu'audience et impose une durée de vie maximale des jetons de 5 minutes (300 secondes). Consultez la the section called "Clés de condition IAM" liste des clés de condition que vous pouvez utiliser pour appliquer les propriétés des jetons.

Exemple de politique IAM

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "sts:IdentityTokenAudience": "https://api.example.com"
                },
                "NumericLessThanEquals": {
                    "sts:DurationSeconds": 300
                }
            }
        }
    ]
}
```

Demandez votre premier jeton Web JSON (JWT)

Vous pouvez demander un jeton Web JSON à l'aide de l'GetWebIdentityTokenAPI. Vous pouvez spécifier les paramètres suivants lorsque vous appelez l'API :

- Public (obligatoire) : destinataire prévu du jeton. Cette valeur renseigne la valeur « aud » dans le JWT. Des services externes valident cette affirmation pour s'assurer que le jeton leur était destiné.
- SigningAlgorithm (obligatoire) : algorithme cryptographique utilisé pour signer le jeton. Les valeurs valides sont ES384 et RS256. Utilisez l'ES384 pour une sécurité et des performances optimales, ou le RS256 pour une compatibilité accrue avec les systèmes qui ne prennent pas en charge l'ECDSA.
- DurationSeconds (facultatif) : durée de vie du jeton en secondes. Les valeurs valides sont comprises entre 60 et 3 600. La valeur par défaut est 300 (5 minutes). Nous recommandons de raccourcir la durée de vie des jetons pour une sécurité accrue.
- Tags (facultatif) : liste de paires clé-valeur à inclure en tant que revendications personnalisées dans le jeton. Les services externes peuvent utiliser ces allégations pour affiner l'autorisation.

L'API renvoie les champs suivants :

- IdentityToken: Le JWT signé sous forme de chaîne codée en base64url. Incluez ce jeton dans les demandes adressées à des services externes.
- Expiration : horodatage UTC de l'expiration du jeton.

Utilisation de AWS INTERFACE DE LIGNE DE COMMANDE (CLI)

```
aws sts get-web-identity-token \
    --audience "https://api.example.com" \
    --signing-algorithm ES384 \
    --duration-seconds 300 \
    --tags Key=team,Value=data-engineering \
           Key=environment,Value=production \
           Key=cost-center,Value=analytics
```

Utilisation AWS SDK pour Python

```
import boto3

sts_client = boto3.client('sts')
```

```
response = sts_client.get_web_identity_token(
    Audience=['https://api.example.com'],
    DurationSeconds=300,
    SigningAlgorithm='RS256',
    Tags=[
        {'Key': 'team', 'Value': 'data-engineering'},
        {'Key': 'environment', 'Value': 'production'},
        {'Key': 'cost-center', 'Value': 'analytics'}
    ]
)

token = response['WebIdentityToken']
```

Vous pouvez également décoder le JWT pour inspecter son contenu à l'aide de bibliothèques JWT standard telles que PyJWT, Python-jose pour Python, Nimbus JOSE+JWT pour Java ou de débogueurs tels que jwt.io. Reportez-vous à the section called "Comprendre les revendications relatives aux jetons" pour plus d'informations sur les réclamations incluses dans le jeton.

Utiliser le jeton avec un service externe

Après avoir reçu le jeton, incluez-le dans les demandes adressées au service externe. La méthode varie selon les services, mais la plupart des services acceptent les jetons dans l'en-tête Authorization. Le service externe doit implémenter une logique de validation des jetons qui extrait les clés JWKS depuis le point de terminaison bien connu de votre émetteur, vérifie la signature du jeton et valide les revendications essentielles avant d'accorder l'accès à vos charges de travail. AWS

Récupérez les clés de vérification et les métadonnées depuis les points de terminaison OpenID Connect (OIDC)

L'URL d'émetteur unique de votre AWS compte héberge les points de terminaison de découverte OpenID Connect (OIDC) qui contiennent les clés de vérification et les métadonnées nécessaires à la vérification des jetons.

L'URL du point de terminaison de découverte OIDC contient des métadonnées que certains fournisseurs utilisent pour vérifier les jetons. Il est disponible à l'adresse suivante :

```
{issuer_url}/.well-known/openid-configuration
```

Le point de terminaison JWKS (JSON Web Key Set) contient des clés utilisées pour vérifier les signatures de jetons. Il est disponible à l'adresse suivante :

```
{issuer_url}/.well-known/jwks.json
```

Récupérez JWKS à l'aide de curl

```
curl https://{issuer_url}/.well-known/jwks.json
```

Réponse :

```
{
  "keys": [
    {
      "kty": "EC",
      "use": "sig",
      "kid": "key-id-1",
      "alg": "ES384",
      "crv": "P-384",
      "x": "base64-encoded-x-coordinate",
      "y": "base64-encoded-y-coordinate"
    },
    {
      "kty": "RSA",
      "use": "sig",
      "kid": "key-id-2",
      "n": "base64-encoded-modulus",
      "e": "AQAB"
    }
  ]
}
```

Utilisation AWS SDK pour Python

```
import requests

# Fetch Openid Configuration
open_id_config_response = requests.get("https://{issuer_url}/.well-known/openid-
configuration")
open_id_config = open_id_config_response.json()

# Fetch JWKS
jwks_response = requests.get("https://{issuer_url}/.well-known/jwks.json")
jwks = jwks_response.json()
```

Nous vous recommandons de mettre ces clés en cache pour éviter de les récupérer à chaque vérification de jeton.

Validations essentielles des allégations

- Objet (sous) : vérifiez que l'objet de la réclamation contient le modèle d'ARN principal IAM attendu.
- Expiration (exp) : assurez-vous que le jeton n'a pas expiré. Les bibliothèques JWT gèrent généralement cela automatiquement.
- Audience (aud) : vérifiez que l'audience correspond à la valeur attendue. Cela empêche l'utilisation de jetons destinés à d'autres services avec le vôtre.
- Émetteur (iss) : vérifiez que l'émetteur correspond au ou aux AWS comptes auxquels vous faites confiance. Tenez à jour une liste d'URL d'émetteurs fiables.

Dans la mesure du possible, vous devez valider des déclarations AWS spécifiques supplémentaires afin de mettre en œuvre un contrôle d'accès précis dans votre service externe. Par exemple, validez la revendication org_id pour restreindre l'accès aux principaux IAM de votre AWS organisation, cochez principal_tags pour appliquer le contrôle d'accès basé sur les attributs (par exemple, autoriser uniquement les environnements de production ou des équipes spécifiques), ou vérifiez les revendications de contexte de session telles que lambda_source_function_arn ou ec2_instance_source_vpc pour restreindre l'accès en fonction de la ressource de calcul. Reportez-vous à la section Comprendre les revendications relatives aux jetons pour obtenir la liste complète des revendications incluses dans le jeton.

Comprendre les revendications relatives aux jetons

Lorsque vous appelez l'GetWebIdentityTokenAPI, AWS Security Token Service renvoie un jeton Web JSON (JWT) signé qui contient un ensemble de revendications représentant l'identité du principal IAM. Ces jetons sont conformes à la norme RFC 7519. Comprendre la structure et le contenu de ces jetons vous permet de mettre en œuvre des flux d'authentification sécurisés, de configurer les validations de revendications appropriées dans les services externes et d'utiliser efficacement les revendications personnalisées pour un contrôle d'accès précis.

Le JWT inclut des déclarations OpenID Connect (OIDC) standard telles que sujet (« sub »), audience (« aud »), émetteur (« iss ») afin de faciliter l'interopérabilité entre différents services externes. AWS STS remplit le jeton avec des revendications AWS spécifiques à l'identité (telles que les balises AWS Account ID et Principal) et des revendications contextuelles de session (comme les ARN d'instance EC2), le cas échéant. Vous pouvez également ajouter des revendications personnalisées au jeton en

les transmettant sous forme de balises de demande à l'GetWebIdentityTokenAPI. Les revendications AWS spécifiques à l'identité, les revendications de contexte de session et les revendications personnalisées sont imbriquées dans l'espace de noms « https://sts.amazonaws.com/ » du jeton.

Reportez-vous à l'exemple de jeton ci-dessous pour obtenir la liste des réclamations incluses dans le jeton. Veuillez noter que toutes ces revendications peuvent ne pas être présentes simultanément dans un jeton.

```
{
  "iss": "https://abc123-def456-ghi789-jkl012.tokens.sts.global.api.aws",
  "aud": "https://api.example.com",
  "sub": "arn:aws:iam::123456789012:role/DataProcessingRole",
  "iat": 1700000000,
  "exp": 1700000900,
  "jti": "xyz123-def456-ghi789-jkl012",
  "https://sts.amazonaws.com/": {
    "aws_account": "123456789012",
    "source_region": "us-east-1",
    "org_id": "o-abc1234567",
    "ou_path": "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/",
    "principal_tags": {
      "environment": "production",
      "team": "data-engineering",
      "cost-center": "engineering"
    },
    "lambda_source_function_arn": "arn:aws:lambda:us-
east-1:123456789012:function:process-data",
    "request_tags": {
        "job-id": "job-2024-001",
        "priority": "high",
        "data-classification": "sensitive"
    }
  }
}
```

Réclamations standard

Les revendications OIDC standard présentes dans les jetons facilitent l'interopérabilité avec un large éventail de services externes. Ces affirmations peuvent être validées à l'aide de la plupart des bibliothèques JWT.

Demander	Nom	Description	Exemple de valeur
bise	Emetteur	URL de l'émetteur spécifique à votre compte. Des services externes valident cette affirmation pour s'assurer qu'elle correspond à leur émetteur de confiance.	https://abc123-def456-ghi789-jkl012.tokens.sts.global.api.aws
aud	Public ciblé	Destinataire prévu pour le jeton spécifié dans la GetWebIdentityToken demande.	https://api.example.com
sub	Sujet	L'ARN du principal IAM qui a demandé le jeton.	arn:aws:iam : :123456789012 : role/DataProcessingRole
iat	Publié à	NumericDate valeur qui identifie l'heure à laquelle le JWT a été émis.	1700000000
exp	Expiration	NumericDate valeur qui identifie le délai d'expiration à partir duquel le JWT NE DOIT PAS être accepté pour traitement.	1700000900
jti	IDENTIFIANT JET	Identifiant unique pour cette instance de jeton.	xyz123-def456-ghi789-jkl012

Réclamations personnalisées

En plus des revendications OIDC standard, AWS STS ajoute des revendications concernant l'identité et le contexte de session, le cas échéant. Vous pouvez également ajouter vos propres revendications au jeton en les transmettant sous forme de balises de demande. Les revendications personnalisées sont imbriquées dans l'espace de https://sts.amazonaws.com/ noms.

AWS demandes d'identité

Ces réclamations fournissent des informations détaillées sur votre AWS compte, la structure de votre organisation et le responsable IAM.

Demander	Description	Cartes vers la clé de condition	Exemple de valeur
compte_aws	Votre identifiant AWS de compte	aws : Principal Account	123456789012
source_region	La AWS région où le jeton a été demandé	aws : Requested Region	us-east-1
id_org	L'identifiant de votre AWS organisation (si votre compte fait partie d'une organisation)	aws : PrincipalOrg ID	o-abc1234567
ou_path	Le parcours de votre unité organisationnelle (le cas échéant)	aws : Principal OrgPaths	o- a1b2c3d4e5/r - ab12/ou -ab12- - ab12-22222222/ 11111111/ou
tags_principaux	Tags attachés à la session IAM principale ou à la session de rôle assumé. Lorsqu'un jeton est demandé alors que le principal IAM demandeur possède à la fois des balises principales et des balises de session, les balises de session seront présentes dans le JWT.	annonces :Principa lTag/<tag-key>	{"environnement » : « production », « équipe » : « ingénierie des données », « centre de coûts » :"ingénierie "}

Déclarations relatives au contexte de session

Ces allégations fournissent des informations sur l'environnement informatique et la session d'où provient la demande de jeton. AWS STS inclut automatiquement ces réclamations, le cas échéant, en fonction du contexte de session du principal demandeur.

Demander	Description	Cartes vers la clé de condition	Exemple de valeur
exp de session d'origine	Date à laquelle les informations d'identification de session de rôle d'origine expireront (pour les rôles supposés)	N/A	15/01/2024 T10:00:00Z
fournisseur_fédéré	Le nom du fournisseur d'identité pour les sessions fédérées	aws : Federated Provider	arn:aws:iam : :111122223333:oidc-provider/your_oidc_provider
identity_store_user_id	ID utilisateur IAM Identity Center	boutique d'identité : UserId	utilisateur-abc123def456
identity_store_arn	ARN de la boutique d'identités Identity Center	boutique d'identité : IdentityStoreArn	arn:aws : identitystore : :123456789012 : -abc1234567 identitystore/d
ec2_source_instance_arn	ARN de l'instance EC2 demandeuse	EC2 : SourceInstanceArn	arn:aws:ec2:us-east- 1:123456789012 : -abc123def456 instance/i
ec2_instance_source_vpc	ID VPC où les informations d'identification du rôle EC2 ont été fournies	aws : Ec2InstanceSourceVpc	vpc-abc123def456
ec2_instance_source_privé_ipv4	Adresse IPv4 privée de l'instance EC2	aws : Ec2InstanceSourcePrivateIPv4	10,0.1,25
ec2_role_delivery	Version du service de métadonnées d'instance	EC2 : RoleDelivery	2

Demander	Description	Cartes vers la clé de condition	Exemple de valeur
identité_source	Identité de la source définie par le principal	aws : SourceIdentity	administrateur-utilisateur
lambda_source_function_arn	ARN de la fonction Lambda appelante	lambda : SourceFunctionArn	arn:aws:lambda:us-east- 1:123456789012:function:my-function
service d'émission d'informations d'identification de la colle	AWS Identifiant du service Glue pour les tâches Glue	colle : CredentialIssuingService	glue.amazonaws.com

Tags de demande

Vous pouvez ajouter des revendications personnalisées aux jetons en spécifiant des balises dans la demande GetWebIdentityToken d'API. Ces revendications apparaissent dans le champ request_tags du jeton et vous permettent de transmettre des informations spécifiques que les services externes peuvent utiliser pour prendre des décisions d'autorisation précises. Vous pouvez spécifier jusqu'à 50 balises par demande.

Exemple de demande :

```
response = sts_client.get_web_identity_token(
    Audience=['https://api.example.com'],
    SigningAlgorithm='ES384',
    Tags=[
        {'Key': 'team', 'Value': 'data-engineering'},
        {'Key': 'cost-center', 'Value': 'analytics'},
        {'Key': 'environment', 'Value': 'production'}
    ]
)
```

Réclamations résultantes sous forme de jeton :

```
{
  "request_tags": {
    "team": "data-engineering",
    "cost-center": "analytics",
    "environment": "production"
  }
}
```

Contrôler l'accès à l'aide de politiques IAM

IAM propose plusieurs types de politiques pour contrôler l'accès à la fonctionnalité de fédération d'identités sortantes. Vous pouvez utiliser des politiques basées sur l'identité pour contrôler quels responsables IAM peuvent demander des jetons et appliquer des propriétés de jetons spécifiques telles que l'audience, les durées de vie et les algorithmes de signature. Les politiques de contrôle des services (SCP) vous permettent d'appliquer des restrictions à l'échelle de l'organisation en matière de génération de jetons pour tous les comptes de vos organisations. AWS Les politiques de contrôle des ressources (RCP) contrôlent l'accès au niveau des ressources. Vous pouvez également utiliser les politiques relatives aux terminaux VPC pour limiter les principaux accès à l' AWS STS `GetWebIdentityToken`API via vos points de terminaison VPC, en ajoutant des contrôles au niveau du réseau à votre niveau de sécurité. Cette section explique comment mettre en œuvre des contrôles d'accès précis à l'aide de ces types de politiques et de ces clés de condition.

Pour demander des jetons d'identité, un responsable IAM doit disposer de l'`sts:GetWebIdentityToken`autorisation. Accordez cette autorisation par le biais de politiques d'identité associées aux utilisateurs ou aux rôles IAM. Pour autoriser la transmission de balises (paires clé/valeur) à l' GetWebIdentityToken appel, le principal IAM doit disposer de cette `sts:TagGetWebIdentityToken` autorisation.

- Utilisez la clé de IdentityTokenAudience condition sts : pour limiter les services externes qui peuvent recevoir des jetons.
- Utilisez la clé de DurationSeconds condition sts : pour appliquer la durée de vie maximale des jetons.
- Utilisez la clé de SigningAlgorithm condition sts : pour demander des algorithmes cryptographiques spécifiques.
- Utilisez la clé de RequestTag condition aws : comparez la paire clé-valeur de balise qui a été transmise dans la demande avec la paire de balises que vous spécifiez dans la politique.

- Utilisez la clé de TagKeys condition aws : pour comparer les clés de balise d'une demande avec les clés que vous spécifiez dans la politique.

Reportez-vous à la section IAM et aux clés de AWS STS condition pour en savoir plus sur les clés de condition disponibles dans les politiques IAM.

Cet exemple de politique d'identité combine plusieurs clés de condition :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowTokenGenerationWithRestrictions",
            "Effect": "Allow",
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "sts:IdentityTokenAudience": [
                        "https://api1.example.com",
                        "https://api2.example.com"
                    ]
                },
                "NumericLessThanEquals": {
                    "sts:DurationSeconds": 300
                },
                "StringEquals": {
                    "sts:SigningAlgorithm": "ES384"
                }
            }
        }
    ]
}
```

Bonnes pratiques

Suivez ces recommandations pour fédérer en toute sécurité vos AWS identités à des services externes.

- Utilisez des durées de vie courtes pour les jetons : demandez des jetons dont la durée de vie est la plus courte pour répondre à vos besoins opérationnels.

- Implémentez l'accès avec le moindre privilège et limitez les propriétés des jetons à l'aide de politiques IAM : accordez l'`sts:GetWebIdentityToken`autorisation uniquement aux principaux administrateurs IAM qui en ont besoin. Utilisez des clés de condition pour spécifier les algorithmes de signature, les audiences de jetons autorisées et la durée de vie maximale des jetons, selon vos besoins.
- Validez les déclarations dans des services externes : pour des raisons de sécurité, validez toujours les déclarations pertinentes telles que le sujet (« sous »), l'audience (« aud »), etc. pour vous assurer qu'elles correspondent aux valeurs attendues. Validez les demandes personnalisées dans la mesure du possible afin de permettre des décisions d'autorisation précises dans les services externes.

Informations d'identification de sécurité temporaires dans IAM

Vous pouvez utiliser le AWS Security Token Service (AWS STS) pour créer et fournir à des utilisateurs de confiance des informations d'identification de sécurité temporaires qui peuvent contrôler l'accès à vos AWS ressources. Les informations d'identification temporaires fonctionnent de manière presque identique aux informations d'identification des clés d'accès à long terme, à quelques différences près :

- Les informations d'identification de sécurité temporaires sont à court terme, comme leur nom l'indique. Elles peuvent être configurées pour être valides de quelques minutes à plusieurs heures. Une fois les informations d'identification AWS expirées, il ne les reconnaît plus ou n'autorise aucun type d'accès à partir des demandes d'API effectuées avec eux.
- Les informations d'identification de sécurité temporaires ne sont pas stockées avec l'utilisateur, mais sont générées automatiquement et fournies à l'utilisateur sur demande. Lorsque (ou même avant) les informations d'identification de sécurité temporaires arrivent à expiration, l'utilisateur peut en demander de nouvelles, tant qu'il y est autorisé.

Par conséquent, les informations d'identification temporaires présentent les avantages suivants par rapport aux informations d'identification à long terme :

- Vous n'êtes pas obligé de distribuer ou d'intégrer des informations AWS de sécurité à long terme dans une application.
- Vous pouvez donner accès à vos AWS ressources aux utilisateurs sans avoir à leur définir une AWS identité. Les informations d’identification temporaires sont le fondement des rôles et de la fédération d’identité.

- Les informations d'identification de sécurité temporaires ont une durée de vie limitée. Il n'est donc pas nécessaire de les mettre à jour ou de les révoquer explicitement lorsqu'elles deviennent obsolètes. Une fois que les informations d'identification de sécurité temporaires arrivent à expiration, elles ne peuvent pas être réutilisées. Vous pouvez spécifier le délai de validité des informations d'identification, jusqu'à une certaine limite.

AWS STS and AWS régions

Les informations d'identification de sécurité temporaires sont générées par AWS STS. Par défaut, AWS STS il s'agit d'un service mondial avec un seul point de terminaison à`https://sts.amazonaws.com`. Cependant, vous pouvez également choisir d'effectuer des appels d' AWS STS API vers des terminaux situés dans n'importe quelle autre région prise en charge. Cela permet de réduire la latence (décalage serveur) en envoyant des demandes aux serveurs situés dans une région géographiquement plus proche de vous. Quelle que soit la région d'où proviennent vos informations d'identification, elles fonctionnent dans le monde entier. Pour plus d'informations, veuillez consulter Gérer AWS STS dans un Région AWS.

Scénarios courants d'informations d'identification temporaires

Les informations d'identification temporaires sont utiles dans des scénarios impliquant la fédération d'identité, la délégation, l'accès entre comptes et les rôles IAM.

Fédération des identités

Vous pouvez gérer les identités de vos utilisateurs dans un système externe AWS et autoriser les utilisateurs qui se connectent à partir de ces systèmes à effectuer des AWS tâches et à accéder à vos AWS ressources. IAM prend en charge deux types de fédération d'identité. Dans les deux cas, les identités sont stockées en dehors de AWS. La distinction réside dans l'endroit où se trouve le système externe : dans votre centre de données ou dans un tiers externe sur le Web. Pour comparer les fonctionnalités d'informations d'identification de sécurité temporaires pour la fédération d'identité, consultez Comparez les AWS STS informations d'identification.

Pour plus d'informations sur les fournisseurs d'identité externes, consultez Fournisseurs d'identité et fédération au sein AWS.

- Fédération OpenID Connect (OIDC) — Vous pouvez autoriser les utilisateurs à se connecter à l'aide d'un fournisseur d'identité tiers bien connu, tel que Login with Amazon, Facebook, Google ou tout autre OIDC-compatible fournisseur. Pour votre application mobile ou Web, vous n'avez pas

besoin de créer de code de connexion personnalisé ni de gérer vos propres identités d'utilisateur. L'utilisation de la fédération OIDC vous aide à Compte AWS garantir votre sécurité, car vous n'avez pas à distribuer des informations d'identification de sécurité à long terme, telles que des clés d'accès utilisateur IAM, avec votre application. Pour de plus amples informations, veuillez consulter Fédération OIDC.

AWS STS La fédération OIDC prend en charge la connexion avec Amazon, Facebook, Google et tout fournisseur d'identité compatible avec OpenID Connect (OIDC).

> **Note**
>
> Pour les applications mobiles, nous vous recommandons d'utiliser Amazon Cognito. Vous pouvez utiliser ce service avec des AWS kits SDK pour le développement mobile afin de créer des identités uniques pour les utilisateurs et de les authentifier pour un accès sécurisé à vos AWS ressources. Amazon Cognito prend en charge les mêmes fournisseurs d'identité que AWS STS, prend également en charge l'accès non authentifié (invité) et vous permet de migrer les données des utilisateurs lorsqu'ils se connectent. Amazon Cognito fournit également des opérations d'API pour la synchronisation des données utilisateur afin de les conserver à mesure que les utilisateurs passent d'un périphérique à l'autre. Pour plus d'informations, consultez Authentification avec Amplify dans la documentation Amplify.

- Fédération SAML : vous pouvez authentifier les utilisateurs du réseau de votre organisation, puis leur donner accès AWS sans créer de nouvelles AWS identités pour eux et les obliger à se connecter avec des informations de connexion différentes. C'est ce que l'on appelle l'approche d'authentification unique pour l'accès temporaire. AWS STS prend en charge des normes ouvertes telles que le langage SAML (Security Assertion Markup Language) 2.0, qui vous permet d'utiliser Microsoft AD FS pour tirer parti de votre Microsoft Active Directory. SAML 2.0 vous permet également de gérer votre propre solution pour fédérer les identités des utilisateurs. Pour de plus amples informations, veuillez consulter Fédération SAML 2.0.
  - Courtier de fédération personnalisé : vous pouvez utiliser le système d'authentification de votre organisation pour autoriser l'accès aux AWS ressources. Pour obtenir un exemple de scénario, consultez Activez l'accès personnalisé du courtier d'identité au AWS console.
  - Federation using SAML 2.0 (Fédération à l'aide de SAML 2.0) : vous pouvez utiliser le système d'authentification de votre organisation et SAML pour accorder l'accès aux ressources AWS . Pour obtenir des informations et un exemple de scénario, consultez Fédération SAML 2.0.

Rôles pour l'accès entre comptes

De nombreuses organisations gèrent plusieurs Compte AWS. À l'aide des rôles et de l'accès entre comptes, vous pouvez définir des identités utilisateur dans un compte et utiliser ces identités pour accéder aux ressources AWS dans d'autres comptes qui appartiennent à votre organisation. Cette procédure s'appelle la délégation pour un accès temporaire. Pour de plus amples informations sur la création de rôles entre comptes, veuillez consulter la rubrique Créer un rôle pour attribuer des autorisations à un utilisateur IAM. Pour savoir si les principaux des comptes situés en dehors de votre zone de confiance (organisation ou compte de confiance) ont accès à vos rôles, veuillez consulter Qu'est-ce que l'Analyseur d'accès IAM ?.

Rôles pour Amazon EC2

Si vous exécutez des applications sur des instances Amazon EC2 et que ces applications doivent accéder à des ressources AWS , vous pouvez fournir des informations d'identification de sécurité temporaires à vos instances lorsque vous les lancez. Ces informations d'identification de sécurité temporaires sont disponibles pour toutes les applications qui s'exécutent sur l'instance, vous n'avez donc pas besoin de stocker des informations d'identification à long terme sur l'instance. Pour de plus amples informations, veuillez consulter Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2.

Pour en savoir plus sur les informations d'identification des rôles IAM Amazon EC2, consultez Rôles IAM pour Amazon EC2 dans le Guide de l'utilisateur Amazon Elastic Compute Cloud.

Autre AWS services

Vous pouvez utiliser des informations d'identification de sécurité temporaires pour accéder à la plupart AWS des services. Pour obtenir une liste des services qui acceptent les informations d'identification de sécurité temporaires, consultez la section AWS services qui fonctionnent avec IAM.

Exemples d'application qui utilisent des informations d'identification temporaires

Vous pouvez utiliser AWS Security Token Service (AWS STS) pour créer et fournir à des utilisateurs de confiance des informations d'identification de sécurité temporaires qui peuvent contrôler l'accès à vos AWS ressources. Pour plus d'informations sur AWS STS, consultezInformations d'identification de sécurité temporaires dans IAM. Pour savoir comment gérer les informations d'identification AWS STS de sécurité temporaires, vous pouvez télécharger les exemples d'applications suivants qui mettent en œuvre des exemples de scénarios complets :

- Activation de la fédération pour AWS utiliser Windows Active Directory, ADFS et SAML 2.0. Montre comment déléguer l'accès via la fédération d'entreprise à AWS Windows Active Directory (AD), Active Directory Federation Services (ADFS) 2.0 et SAML (Security Assertion Markup Language) 2.0.
- Activez l'accès personnalisé du courtier d'identité au AWS consoleExplique comment créer un proxy de fédération permettant d'autoriser l'authentification unique (SSO) afin que les utilisateurs Active Directory existants puissent se connecter à Console de gestion AWS.
- Comment utiliser Shibboleth pour Sign-On Single to the. Console de gestion AWS. Explique comment utiliser Shibboleth et SAML pour fournir aux utilisateurs un accès avec authentification unique (SSO) à l' Console de gestion AWS.

Exemples pour la fédération OIDC

Les exemples d'applications suivants montrent comment utiliser la fédération OIDC avec des fournisseurs tels que Login with Amazon, Amazon Cognito, Facebook ou Google. Vous pouvez échanger l'authentification auprès de ces fournisseurs contre des informations AWS de sécurité temporaires pour accéder aux AWS services.

- Tutoriels Amazon Cognito — Nous vous recommandons d'utiliser Amazon Cognito avec les AWS kits SDK pour le développement mobile. Amazon Cognito offre le moyen le plus simple de gérer l'identité pour les applications mobiles et fournit des fonctions supplémentaires telles que la synchronisation et l'identité entre périphériques. Pour plus d'informations sur Amazon Cognito, consultez Authentification avec Amplify dans la documentation Amplify.

Ressources supplémentaires pour les informations d'identification de sécurité temporaires

Les scénarios et applications suivants peuvent vous aider à utiliser les informations d'identification de sécurité temporaires :

- Comment intégrer AWS STS SourceIdentity votre fournisseur d'identité. Cet article explique comment configurer l' AWS STS `SourceIdentity`attribut lorsque vous utilisez Okta, Ping ou en OneLogin tant qu'IdP.
- Fédération OIDC. Cette section explique comment configurer les rôles IAM lorsque vous utilisez la fédération OIDC et l'API `AssumeRoleWithWebIdentity`.

- Accès sécurisé aux API avec MFA. Cette rubrique explique comment utiliser les rôles pour exiger l'authentification multi-facteur (MFA) afin de protéger les actions d'API sensibles de votre compte.

Pour plus d'informations sur les politiques et les autorisations, AWS consultez les rubriques suivantes :

- Gestion des accès pour AWS resources
- Logique d'évaluation de politiques.
- Managing Access Permissions to Your Amazon S3 Resources (gestion des autorisations d'accès à vos ressources Amazon S3) dans le guide de l'utilisateur du service de stockage simple Amazon.
- Pour savoir si les principaux des comptes situés en dehors de votre zone de confiance (organisation ou compte de confiance) ont accès à vos rôles, veuillez consulter Qu'est-ce que l'Analyseur d'accès IAM ?.

Comparez les AWS STS informations d'identification

Le tableau suivant compare les fonctionnalités des opérations d'API AWS STS qui renvoient des informations d'identification de sécurité temporaires. Pour en savoir plus sur les différentes méthodes disponibles pour demander des identifiants de sécurité temporaires en endossant un rôle, consultez Méthodes pour assumer un rôle. Pour en savoir plus sur les différentes opérations d' AWS STS API qui vous permettent de transmettre des balises de session, consultezTransmettre les balises de session AWS STS.

> **Note**
>
> Vous pouvez envoyer des appels AWS STS d'API soit à un point de terminaison global, soit à l'un des points de terminaison régionaux. En choisissant un point de terminaison plus proche de vous, il est possible de réduire la latence et d'améliorer les performances de vos appels d'API. Vous pouvez également diriger vos appels vers un autre point de terminaison régional si vous n'êtes plus en mesure de communiquer avec le point de terminaison initial. Si vous utilisez l'une des différentes méthodes AWS SDKs, utilisez cette méthode du SDK pour spécifier une région avant d'effectuer l'appel d'API. Si vous créez manuellement des demandes d'API HTTP, vous devez les diriger vous-même vers le point de terminaison approprié. Pour plus d'informations, consultez la section AWS STS de Régions et points de terminaison et Gérer AWS STS dans un Région AWS.

AWS STS API	Qui peut appeler	Durée de vie des informations d'identification (min | max | par défaut)	Prise en charge de l'authentification MFA[1]	Prise en charge de la politique de session[2]	Restrictions applicables aux informations d'identification temporaires générées
AssumeRole	Utilisateur IAM ou rôle IAM disposant d'informations d'identification de sécurité temporaires existantes	15 min | Durée de session maximale | 1 h	Oui	Oui	Impossible d'appeler `GetFederationToken` ou `GetSessionToken` .
AssumeRoleWithSAML	N'importe quel utilisateur ; le principal doit transmettre une réponse d'authentification SAML qui spécifie l'authentification par un fournisseur d'identité reconnu	15 min | Durée de session maximale | 1 h	Non	Oui	Impossible d'appeler `GetFederationToken` ou `GetSessionToken` .
AssumeRoleWithWebIdentity	Tout utilisateur ; l'appelant doit transmettre un jeton JWT conforme à l'OIDC qui atteste d'une authentification	15 min | Durée de session maximale | 1 h	Non	Oui	Impossible d'appeler `GetFederationToken` ou `GetSessionToken` .

AWS STS API	Qui peut appeler	Durée de vie des informations d'identification (min | max | par défaut)	Prise en charge de l'authentification MFA[1]	Prise en charge de la politique de session[2]	Restrictions applicables aux informations d'identification temporaires générées
	auprès d’un fournisseur d’identité connu				
GetFederationToken	Utilisateur IAM ou Utilisateur racine d'un compte AWS	Utilisateur IAM : 15 min | 36 h | 12 h Utilisateur racine : 15 min | 1 heure | 1 heure	Non	Oui	Impossible d'appeler des opérations IAM à l'aide de l' AWS API AWS CLI or. Cette limitation ne s'applique pas aux sessions de console. Impossible d'appeler AWS STS les opérations sauf `GetCallerIdentity` .[4] Connexion avec authentification unique (SSO) à la console autorisée.[5]

AWS STS API	Qui peut appeler	Durée de vie des informations d'identification (min | max | par défaut)	Prise en charge de l'authentification MFA[1]	Prise en charge de la politique de session[2]	Restrictions applicables aux informations d'identification temporaires générées
GetSessionToken	Utilisateur IAM ou Utilisateur racine d'un compte AWS	Utilisateur IAM : 15 min | 36 h | 12 h Utilisateur racine : 15 min | 1 heure | 1 heure	Oui	Non	Impossible d'appeler les opérations d'API IAM si les informations MFA ne figurent pas dans la demande. Impossible d'appeler les opérations d' AWS STS API sauf `AssumeRole` ou`GetCallerIdentity` . Connexion avec authentification unique (SSO) à la console non autorisée.[6]

[1] Prise en charge de l'authentification MFA. Vous pouvez inclure des informations sur un dispositif d'authentification multifactorielle (MFA) lorsque vous appelez AssumeRole les opérations GetSessionToken et API. De cette façon, les informations d'identification de sécurité temporaires obtenues lors l'appel d'API peuvent uniquement être utilisées par des utilisateurs authentifiés à l'aide d'un dispositif MFA. Pour plus d'informations, veuillez consulter Accès sécurisé aux API avec MFA.

[2] Prise en charge de la politique de session. Les politiques de session sont des politiques que vous transmettez en paramètre lorsque vous créez par programmation une session temporaire pour un rôle ou une session utilisateur AWS STS fédérée. Cette politique limite les autorisations à partir de la politique basée sur l'identité du rôle ou de l'utilisateur qui sont attribuées à la session. Les autorisations de la session obtenues sont une combinaison des stratégies basées sur l'identité

de l'entité et des stratégies de session. Les politiques de session ne peuvent pas être utilisées pour accorder plus d'autorisations que celles autorisées par la politique basée sur l'identité du rôle endossé actuellement. Pour de plus amples informations sur les autorisations de session de rôle, veuillez consulter Politiques de session.

[3] Paramètre de durée de session maximale. Utilisez le paramètre `DurationSeconds` pour spécifier la durée de la session de votre rôle entre 900 secondes (15 minutes) et la durée de session maximale pour le rôle. Pour savoir comment afficher la valeur maximale pour votre rôle, veuillez consulter Mettre à jour la durée de session maximale pour un rôle.

[4] GetCallerIdentity. Aucune autorisation n'est requise pour effectuer cette opération. Si un administrateur ajoute une politique à votre utilisateur ou rôle IAM qui refuse explicitement l'accès à l'action `sts:GetCallerIdentity`, vous pouvez toujours effectuer cette opération. Les autorisations ne sont pas requises, car les mêmes informations sont renvoyées lorsqu'un utilisateur ou un rôle IAM se voit refuser l'accès. Pour afficher un exemple de réponse, consultez Je ne suis pas autorisé à effectuer : iam : DeleteVirtualMFADevice.

[5] Connexion avec authentification unique (SSO) à la console. Pour prendre en charge l'authentification unique, vous AWS pouvez appeler un point de terminaison de fédération (`https://signin.aws.amazon.com/federation`) et transmettre des informations d'identification de sécurité temporaires. Le point de terminaison retourne un jeton que vous pouvez utiliser pour créer une URL qui connecte directement l'utilisateur à la console, sans avoir besoin d'un mot de passe. Pour plus d'informations, consultez Permettre aux principaux fédérés SAML 2.0 d'accéder au Console de gestion AWS la section Comment activer l'accès multicompte à la console de AWS gestion dans le blog sur la AWS sécurité.

[6] Une fois que vous avez récupéré vos informations d'identification temporaires, vous ne pouvez pas accéder à l'interface Console de gestion AWS en transmettant les informations d'identification au point de terminaison d'authentification unique de fédération. Pour de plus amples informations, veuillez consulter Activez l'accès personnalisé du courtier d'identité au AWS console.

Jeton porteur de service

Certains AWS services nécessitent que vous soyez autorisé à obtenir un jeton de support de AWS STS service avant de pouvoir accéder à leurs ressources par programmation. Ces services prennent en charge un protocole qui vous oblige à utiliser un jeton porteur au lieu d'utiliser une AWS Signature Version 4 pour les demandes d'API traditionnelle. Lorsque vous effectuez des opérations AWS CLI ou des opérations d' AWS API qui nécessitent des jetons porteurs, le AWS service demande un jeton

porteur en votre nom. Le service vous fournit le jeton, que vous pouvez ensuite utiliser pour effectuer toute opération ultérieure dans ce service.

AWS STS les jetons de support incluent des informations issues de votre authentification principale d'origine qui peuvent affecter vos autorisations. Ces informations peuvent comprendre des balises de principal ou de session, ainsi que des politiques de session. L'ID de clé d'accès du jeton commence par le préfixe `ABIA`. Cela vous aide à identifier les opérations effectuées à l'aide de jetons porteurs de service dans les journaux CloudTrail.

> **Important**
>
> Le jeton porteur ne peut être utilisé que pour les appels vers le service qui le génère et dans la région dans laquelle il a été généré. Vous ne pouvez pas l'utiliser pour effectuer des opérations dans d'autres services ou régions.

Un exemple de service qui prend en charge les jetons au porteur est AWS CodeArtifact. Avant de pouvoir interagir avec AWS CodeArtifact un gestionnaire de packages tel que NPM, Maven ou PIP, vous devez appeler l'opération. `aws codeartifact get-authorization-token` Cette opération renvoie un jeton porteur que vous pouvez utiliser pour effectuer des AWS CodeArtifact opérations. Si vous préférez, vous pouvez aussi utiliser la commande `aws codeartifact login` qui exécute la même opération, puis configure automatiquement votre client.

Si vous effectuez une action dans un AWS service qui génère un jeton porteur pour vous, vous devez disposer des autorisations suivantes dans votre politique IAM :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowServiceBearerToken",
            "Effect": "Allow",
            "Action": "sts:GetServiceBearerToken",
            "Resource": "*"
        }
    ]
}
```

Pour obtenir un exemple de jeton de support de service, consultezUtilisation de stratégies basées sur l'identité pour AWS CodeArtifact dans le Guide de l'utilisateur AWS CodeArtifact.

Demande d'identifiants de sécurité temporaires

Pour demander des informations d'identification de sécurité temporaires, vous pouvez utiliser les opérations AWS Security Token Service (AWS STS) dans l' AWS API. Il s'agit notamment des opérations visant à créer et à fournir aux utilisateurs de confiance des informations d'identification de sécurité temporaires qui peuvent contrôler l'accès à vos AWS ressources. Pour plus d'informations sur AWS STS, consultezInformations d'identification de sécurité temporaires dans IAM. Pour en savoir plus sur les différentes méthodes disponibles pour demander des informations d'identification de sécurité temporaires en endossant un rôle, consultez Méthodes pour assumer un rôle.

Pour appeler les opérations d'API, vous pouvez utiliser l'un des kits SDK AWS. Les kits SDK sont disponibles pour de nombreux langages de programmation et environnements tels que Java, .NET, Python, Ruby, Android et iOS. Ces kits SDK exécutent des tâches telles que la signature cryptographique de vos demandes, les nouvelles tentatives de demande, lorsque cela est nécessaire, ainsi que la gestion des réponses d'erreur. Vous pouvez également utiliser l'API AWS STS Query, qui est décrite dans la référence des AWS Security Token Service API. Enfin, deux outils de ligne de commande prennent en charge les AWS STS commandes : le AWS Command Line Interface, et le AWS Tools for Windows PowerShell.

Les opérations d' AWS STS API créent une nouvelle session avec des informations d'identification de sécurité temporaires qui incluent une paire de clés d'accès et un jeton de session. La paire de clés d'accès comprend un ID de clé d'accès et une clé secrète. Ces informations d'identification permettent aux utilisateurs (ou une application exécutée par l'utilisateur) d'accéder à vos ressources.

Vous pouvez créer une session de rôle et transmettre des politiques de session et des balises de session par programmation à l'aide des opérations d' AWS STS API. Les autorisations de la session obtenues sont une combinaison des politiques basées sur l'identité du rôle et des politiques de session. Pour de amples informations sur les politiques de session, veuillez consulter Politiques de session. Pour de plus amples informations sur les balises de session, veuillez consulter Transmettre les balises de session AWS STS.

> **Note**
>
> La taille du jeton de session renvoyé par les opérations d' AWS STS API n'est pas fixe. Nous vous recommandons fortement de ne pas formuler d'hypothèses quant à sa taille maximale. La taille du jeton est en général inférieure à 4 096 octets, mais elle peut varier.

Utilisation AWS STS avec AWS Régions

Vous pouvez envoyer des appels AWS STS d'API à un point de terminaison global ou à l'un des points de terminaison régionaux. En choisissant un point de terminaison plus proche de vous, il est possible de réduire la latence et d'améliorer les performances de vos appels d'API. Vous pouvez également diriger vos appels vers un autre point de terminaison régional si vous n'êtes plus en mesure de communiquer avec le point de terminaison initial.

Si vous utilisez l'un des différents AWS SDK, utilisez cette méthode SDK pour spécifier une région avant d'effectuer l'appel d'API. Si vous créez manuellement des demandes d'API HTTP, vous devez les diriger vous-même vers le point de terminaison approprié. Pour plus d'informations, consultez la section AWS STS de Régions et points de terminaison et Gérer AWS STS dans un Région AWS.

Voici les opérations d'API que vous pouvez utiliser pour obtenir des informations d'identification temporaires à utiliser dans votre AWS environnement et vos applications.

Demande d'informations d'identification pour la délégation et la fédération entre comptes par l'intermédiaire d'un fournisseur d'identité personnalisé

L'opération `AssumeRole` d'API est utile pour permettre aux utilisateurs IAM existants d'accéder à AWS des ressources auxquelles ils n'ont pas encore accès. Par exemple, l'utilisateur peut avoir besoin d'accéder aux ressources d'un autre Compte AWS. Il est également utile comme moyen d'obtenir temporairement un accès privilégié, par exemple, pour fournir une authentification multi-facteur (MFA). Vous devez appeler cette API à l'aide d'informations d'identification actives. Pour savoir qui peut appeler cette opération, consultez Comparez les AWS STS informations d'identification. Pour plus d'informations, consultez Créer un rôle pour attribuer des autorisations à un utilisateur IAM et Accès sécurisé aux API avec MFA.

Pour demander des justificatifs de sécurité temporaires pour la délégation et la fédération de comptes croisés par l'intermédiaire d'un fournisseur d'identité personnalisé

1. Authentifiez-vous à l'aide de vos informations AWS de sécurité. Cet appel doit être effectué à l'aide d'informations d'identification de sécurité AWS valides.
2. Appelez l'opération `AssumeRole`.

L'exemple suivant illustre une demande et une réponse qui utilisent `AssumeRole`. Cet exemple de demande endosse le rôle `demo` pour la durée spécifiée avec la politique de session, les balises de session et l'ID externe et l'identité source inclus. La session obtenue est nommée `John-session`.

Example Exemple de demande

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=AssumeRole
&RoleSessionName=John-session
&RoleArn=arn:aws:iam::123456789012:role/demo
&Policy=%7B%22Version%22%3A%222012-10-17       %22%2C%22Statement%22%3A%5B%7B%22Sid
%22%3A%20%22Stmt1%22%2C%22Effect%22%3A%20%22Allow%22%2C%22Action%22%3A%20%22s3%3A*
%22%2C%22Resource%22%3A%20%22*%22%7D%5D%7D
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&ExternalId=123ABC
&SourceIdentity=DevUser123
&AUTHPARAMS
```

La valeur de stratégie indiquée dans l'exemple précédent est la URL-encoded version de la stratégie suivante :

JSON

```
{"Version":"2012-10-17",       "Statement":
[{"Sid":"Stmt1","Effect":"Allow","Action":"s3:*","Resource":"*"}]}
```

Le paramètre AUTHPARAMS de l'exemple est un espace réservé pour votre signature. Une signature est l'information d'authentification que vous devez inclure dans les requêtes d'API AWS HTTP. Lors de la création de demandes d'API, nous vous recommandons d'utiliser les kits SDK AWS, car ceux-ci gèrent diverses tâches pour vous, notamment la signature des demandes. Si vous devez créer et signer des demandes d'API manuellement, consultez la section Signer des AWS demandes à l'aide de la version 4 de Signature Référence générale d'Amazon Web Services pour savoir comment signer une demande.

Outre les informations d'identification de sécurité temporaires, la réponse inclut l'Amazon Resource Name (ARN) de l'utilisateur fédéré et la date d'expiration des informations d'identification.

Example Exemple de réponse

```
<AssumeRoleResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<AssumeRoleResult>
<SourceIdentity>DevUser123</SourceIdentity>
<Credentials>
  <SessionToken>
   AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
   LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
   QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
   9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
   +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCR/oLxBA==
  </SessionToken>
  <SecretAccessKey>
   wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-07-15T23:28:33.359Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
<AssumedRoleUser>
  <Arn>arn:aws:sts::123456789012:assumed-role/demo/John</Arn>
  <AssumedRoleId>ARO123EXAMPLE123:John</AssumedRoleId>
</AssumedRoleUser>
<PackedPolicySize>34</PackedPolicySize>
<SessionTokenSize>1428</SessionTokenSize>
<SessionTokenUtilization>34</SessionTokenUtilization>
</AssumeRoleResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</AssumeRoleResponse>
```

> **Note**
>
> Les politiques de session et les balises de session augmentent la taille du jeton de session. Si le jeton dépasse la limite de taille, votre demande échoue, même si votre texte en clair se situe dans les autres limites. L'élément de `SessionTokenUtilization` réponse indique le pourcentage de la limite de taille utilisé par le jeton. L'élément de `PackedPolicySize` réponse est obsolète. Utilisez `SessionTokenUtilization` à la place.

Demande d'informations d'identification par le biais d'un fournisseur OIDC

L'opération d'API `AssumeRoleWithWebIdentity` renvoie un ensemble d'informations d'identification de sécurité temporaires AWS en échange d'un jeton Web JSON (JWT). Cela inclut les fournisseurs d'identité publics, tels que Login with Amazon, Facebook, Google, et les fournisseurs qui émettent des JWT compatibles avec la découverte OpenID Connect (OIDC), tels que GitHub Actions ou Azure Devops. Pour de plus amples informations, veuillez consulter Fédération OIDC.

> **Note**
>
> Les requêtes `AssumeRoleWithWebIdentity` ne sont pas signées et ne nécessitent pas d'informations d'identification AWS .

Demande d'informations d'identification par le biais d'un fournisseur OIDC

1. Appelez l'opération `AssumeRoleWithWebIdentity`.

   Lorsque vous appelez`AssumeRoleWithWebIdentity`, AWS valide le jeton présenté en vérifiant la signature numérique à l'aide des clés publiques mises à disposition via le jeu de clés Web JSON (JWKS) de votre IdP. Si le jeton est valide et que toutes les conditions définies dans la politique de confiance des rôles IAM sont remplies, les informations suivantes vous sont AWS renvoyées :

   - Un ensemble d'informations d'identification de sécurité temporaires. Il s'agit d'un ID de clé d'accès, d'une clé d'accès secrète et d'un jeton de session.
   - L'ID de rôle et l'ARN du rôle endossé.
   - Une valeur `SubjectFromWebIdentityToken` qui contient l'ID utilisateur unique.

2. Votre application peut ensuite utiliser les informations d'identification de sécurité temporaires qui ont été renvoyées dans la réponse pour effectuer des appels AWS d'API. Il s'agit du même processus que pour effectuer un appel d' AWS API avec des informations d'identification de sécurité à long terme. La différence est que vous devez inclure le jeton de session, qui permet de AWS vérifier que les informations d'identification de sécurité temporaires sont valides.

Votre application doit mettre en cache les informations d'identification renvoyées par AWS STS et les actualiser si nécessaire. Si votre application est créée à l'aide d'un AWS SDK, celui-ci dispose de fournisseurs d'informations d'identification capables de gérer l'appel `AssumeRoleWithWebIdentity` et l'actualisation des AWS informations d'identification avant leur expiration. Pour plus d'informations, consultez la section Fournisseurs d'informations d'identification normalisés des AWS SDK et des outils dans le Guide de référence des AWS SDK et des outils.

Demande d'informations d'identification via un fournisseur d'identité SAML 2.0

L'opération d'API `AssumeRoleWithSAML` renvoie un ensemble d'informations d'identification de sécurité temporaires pour les principaux fédérés SAML authentifiés par le système d'identité existant de votre organisation. Les utilisateurs doivent également utiliser SAML 2.0 (Security Assertion Markup Language) pour transmettre les informations d'authentification et d'autorisation à AWS. Cette opération d'API s'avère utile pour les organisations qui ont intégré leurs systèmes d'identité (comme Windows Active Directory ou OpenLDAP) à un logiciel capable de générer des assertions SAML. Cette intégration fournit des informations sur l'identité et les autorisations des utilisateurs (comme Active Directory Federation Services ou Shibboleth). Pour de plus amples informations, veuillez consulter Fédération SAML 2.0.

1. Appelez l'opération `AssumeRoleWithSAML`.

   Il s'agit d'un appel non signé, ce qui signifie que vous n'avez pas besoin d'authentifier les informations AWS de sécurité avant d'effectuer la demande.

   > **Note**
   >
   > Un appel à `AssumeRoleWithSAML` n'est pas signé (chiffré). Par conséquent, vous devez uniquement inclure des politiques de session facultatives si la demande est transmise via un intermédiaire approuvé. Dans ce cas, une personne peut modifier la politique afin de supprimer les restrictions.

2. Lorsque vous appelez`AssumeRoleWithSAML`, AWS vérifie l'authenticité de l'assertion SAML. En supposant que le fournisseur d'identité valide l'assertion, il vous AWS renvoie les informations suivantes :

   - Un ensemble d'informations d'identification de sécurité temporaires. Il s'agit d'un ID de clé d'accès, d'une clé d'accès secrète et d'un jeton de session.
   - L'ID de rôle et l'ARN du rôle endossé.
   - Une valeur `Audience` qui contient la valeur de l'attribut `Recipient` de l'élément `SubjectConfirmationData` de l'assertion SAML.
   - Une valeur `Issuer` qui contient la valeur de l'élément `Issuer` de l'assertion SAML.
   - Un `NameQualifier` élément qui contient une valeur de hachage construite à partir de la `Issuer` valeur, de l' Compte AWS ID et du nom convivial du fournisseur SAML. Lors de la combinaison avec l'élément `Subject`, ils identifient de manière unique le principal fédéré SAML.
   - Un élément `Subject` qui contient la valeur de l'élément `NameID` dans l'élément `Subject` de l'assertion SAML.
   - Un élément `SubjectType` qui indique le format de l'élément `Subject`. La valeur peut être `persistent`, `transient` ou l'URI `Format` complet des éléments `Subject` et `NameID` utilisés dans votre assertion SAML. Pour plus d'informations sur l'attribut `NameID` de l'élément `Format`, consultez Configuration d'assertions SAML pour la réponse d'authentification.
3. Utilisez les informations d'identification de sécurité temporaires renvoyées dans la réponse pour effectuer des appels AWS d'API. Il s'agit du même processus que pour effectuer un appel d' AWS API avec des informations d'identification de sécurité à long terme. La différence est que vous devez inclure le jeton de session, ce qui permet à AWS de vérifier la validité des informations d'identification de sécurité temporaires.

Votre application met en cache les informations d'identification. Par défaut, les informations d'identification expirent au bout d'une heure. Si vous n'utilisez pas l'AmazonSTSCredentialsProvideraction du AWS SDK, c'est à vous et à votre application d'appeler à `AssumeRoleWithSAML` nouveau. Appelez cette opération pour obtenir un nouvel ensemble d'informations d'identification de sécurité temporaires avant l'expiration des anciennes.

Demande d'informations d'identification via un fournisseur d'identité personnalisé

L'opération `GetFederationToken` d'API renvoie un ensemble d'informations d'identification de sécurité temporaires pour les principaux utilisateurs AWS STS fédérés. La différence entre cette API

et `AssumeRole` réside dans le fait que le délai d'expiration par défaut est considérablement plus long (12 heures au lieu d'une heure). De plus, vous pouvez utiliser le paramètre `DurationSeconds` pour spécifier une durée pour que les informations d'identification de sécurité temporaires restent valides. Les informations d'identification qui en résultent sont valables pour la durée spécifiée, entre 900 secondes (15 minutes) et 129 600 secondes (36 heures). La période d'expiration plus longue peut contribuer à réduire le nombre d'appels, AWS car vous n'avez pas besoin de nouvelles informations d'identification aussi souvent.

1. Authentifiez-vous à l'aide des informations d'identification de AWS sécurité de votre utilisateur IAM spécifique. Cet appel doit être effectué à l'aide d'informations d'identification AWS de sécurité valides.
2. Appelez l'opération `GetFederationToken`.

L'appel à `GetFederationToken` retourne des informations d'identification de sécurité temporaires constituées d'un jeton de session, d'une clé d'accès, d'une clé secrète et d'un délai d'expiration. Vous pouvez utiliser `GetFederationToken` si vous souhaiter gérer les autorisations au sein de l'organisation (par exemple, pour octroyer des autorisations à l'aide de l'application proxy).

L'exemple suivant illustre une demande et une réponse qui utilisent `GetFederationToken`. Cet exemple de demande fédère l'utilisateur appelant pour la durée spécifiée avec l'ARN de politique de session et les balises de session. La session obtenue est nommée `Jane-session`.

Example Exemple de demande

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetFederationToken
&Name=Jane-session
&PolicyArns.member.1.arn==arn%3Aaws%3Aiam%3A%3A123456789012%3Apolicy%2FRole1policy
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&AUTHPARAMS
```

L'ARN de politique illustré dans l'exemple précédent inclut l' URL-encoded ARN suivant :

```
arn:aws:iam::123456789012:policy/Role1policy
```

Notez également que le paramètre &AUTHPARAMS dans l'exemple est conçu comme un espace réservé pour les informations d'authentification. Il s'agit de la signature, que vous devez inclure dans les requêtes d'API AWS HTTP. Lors de la création de demandes d'API, nous vous recommandons d'utiliser les kits SDK AWS, car ceux-ci gèrent diverses tâches pour vous, notamment la signature des demandes. Si vous devez créer et signer des demandes d'API manuellement, consultez la section Signer des AWS demandes à l'aide de la version 4 de Signature Référence générale d'Amazon Web Services pour savoir comment signer une demande.

Outre les informations d'identification de sécurité temporaires, la réponse inclut l'Amazon Resource Name (ARN) de l'utilisateur fédéré et la date d'expiration des informations d'identification.

Example Exemple de réponse

```
<GetFederationTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetFederationTokenResult>
<Credentials>
  <SessionToken>
   AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
   LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
   QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
   9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
   +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCEXAMPLE==
  </SessionToken>
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-04-15T23:28:33.359Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE;</AccessKeyId>
</Credentials>
<FederatedUser>
  <Arn>arn:aws:sts::123456789012:federated-user/Jean</Arn>
  <FederatedUserId>123456789012:Jean</FederatedUserId>
</FederatedUser>
<PackedPolicySize>26</PackedPolicySize>
<SessionTokenSize>1105</SessionTokenSize>
<SessionTokenUtilization>26</SessionTokenUtilization>
</GetFederationTokenResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</GetFederationTokenResponse>
```

> **Note**
>
> Les politiques de session et les balises de session augmentent la taille du jeton de session. Si le jeton dépasse la limite de taille, votre demande échoue, même si votre texte en clair se situe dans les autres limites. L'élément de `SessionTokenUtilization` réponse indique le pourcentage de la limite de taille utilisé par le jeton. L'élément de `PackedPolicySize` réponse est obsolète. Utilisez `SessionTokenUtilization` à la place.

AWS vous recommande d'accorder des autorisations au niveau des ressources (par exemple, vous associez une politique basée sur les ressources à un compartiment Amazon S3), vous pouvez omettre le paramètre. `Policy` Toutefois, si vous n'incluez pas de politique pour l'utilisateur principal AWS STS fédéré, les informations d'identification de sécurité temporaires n'accorderont aucune autorisation. Dans ce cas, vous devez utiliser des politiques de ressources pour accorder à l'utilisateur fédéré l'accès à vos ressources AWS .

Par exemple, supposons que votre Compte AWS numéro est 111122223333 et que vous avez un compartiment Amazon S3 auquel vous souhaitez autoriser Susan à accéder. Les informations d'identification de sécurité temporaires de Susan ne comprennent pas de politique pour ce compartiment. Dans ce cas, vous devez vous assurer que le compartiment est doté d'une politique avec un ARN correspondant à celui de Susan, par exemple `arn:aws:sts::111122223333:federated-user/Susan`.

Demande d'informations d'identification temporaires pour les utilisateurs dans des environnements non fiables

L'opération d'API `GetSessionToken` retourne un ensemble d'informations d'identification de sécurité temporaires pour un utilisateur IAM existant. Cela est utile pour renforcer la sécurité, par exemple pour autoriser les AWS demandes uniquement lorsque l'authentification multifacteur est activée pour l'utilisateur IAM. Dans la mesure où les informations d'identification sont temporaires, elles offrent une sécurité améliorée lorsqu'un utilisateur IAM accède à vos ressources par le biais d'un environnement moins sécurisé. Les exemples d'environnements moins sécurisés incluent un périphérique mobile ou un navigateur web.

1. Authentifiez-vous à l'aide des informations d'identification de AWS sécurité de votre utilisateur IAM spécifique. Cet appel doit être effectué à l'aide d'informations d'identification AWS de sécurité valides.
2. Appelez l'opération `GetSessionToken`.

3. `GetSessionToken` retourne des informations d'identification de sécurité temporaires constituées d'un jeton de session, d'un ID de clé d'accès et d'une clé d'accès secrète.

Par défaut, les informations d'identification de sécurité temporaires d'un utilisateur IAM restent valides pendant 12 heures maximum. Mais vous pouvez demander un durée de validité allant de 15 minutes à 36 heures à l'aide du paramètre `DurationSeconds`. Pour des raisons de sécurité, la durée d'un jeton pour un Utilisateur racine d'un compte AWS est limitée à une heure.

L'exemple suivant illustre une demande et une réponse qui utilisent `GetSessionToken`. La réponse inclut également l'heure d'expiration des informations d'identification de sécurité temporaires.

Example Exemple de demande

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=1800
&AUTHPARAMS
```

Le paramètre AUTHPARAMS de l'exemple est un espace réservé pour votre signature. Une signature est l'information d'authentification que vous devez inclure dans les requêtes d'API AWS HTTP. Lors de la création de demandes d'API, nous vous recommandons d'utiliser les kits SDK AWS, car ceux-ci gèrent diverses tâches pour vous, notamment la signature des demandes. Si vous devez créer et signer des demandes d'API manuellement, consultez la section Signer des AWS demandes à l'aide de la version 4 de Signature Référence générale d'Amazon Web Services pour savoir comment signer une demande.

Example Exemple de réponse

```
<GetSessionTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetSessionTokenResult>
<Credentials>
  <SessionToken>
   AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT+FvwqnKwRcOIfrRh3c/L
   To6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/IvU1dYUg2RVAJBanLiHb4IgRmpRV3z
   rkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/AXlzBBko7b15fjrBs2+cTQtp
   Z3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE
  </SessionToken>
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
```

```
  </SecretAccessKey>
  <Expiration>2011-07-11T19:55:29.611Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
<SessionTokenSize>856</SessionTokenSize>
<SessionTokenUtilization>20</SessionTokenUtilization>
</GetSessionTokenResult>
<ResponseMetadata>
<RequestId>58c5dbae-abef-11e0-8cfe-09039844ac7d</RequestId>
</ResponseMetadata>
</GetSessionTokenResponse>
```

La `GetSessionToken` demande peut éventuellement inclure des `TokenCode` valeurs pour la `SerialNumber` vérification de l'authentification AWS multifactorielle (MFA). Si les valeurs fournies sont valides, AWS STS fournit des informations d'identification de sécurité temporaires qui incluent l'état de l'authentification MFA. Les informations d'identification de sécurité temporaires peuvent ensuite être utilisées pour accéder aux opérations de l' MFA-protected API ou aux AWS sites Web tant que l'authentification MFA est valide.

L'exemple suivant illustre une demande `GetSessionToken` qui comprend un code de vérification MFA et un numéro de série du dispositif.

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=7200
&SerialNumber=YourMFADeviceSerialNumber
&TokenCode=123456
&AUTHPARAMS
```

> **Note**
>
> L'appel AWS STS peut être destiné au point de terminaison global ou à l'un des terminaux régionaux que vous Compte AWS activez. Pour plus d'informations, veuillez consulter la section AWS STS de Régions et points de terminaison.
> Le paramètre `AUTHPARAMS` de l'exemple est un espace réservé pour votre signature. Une signature est l'information d'authentification que vous devez inclure dans les requêtes d'API AWS HTTP. Lors de la création de demandes d'API, nous vous recommandons d'utiliser les kits SDK AWS, car ceux-ci gèrent diverses tâches pour vous, notamment la signature des demandes. Si vous devez créer et signer des demandes d'API manuellement, consultez la

section Signer des AWS demandes à l'aide de la version 4 de Signature Référence générale d'Amazon Web Services pour savoir comment signer une demande.

Utilisez des informations d'identification temporaires avec AWS resources

Vous pouvez utiliser des informations d'identification de sécurité temporaires pour effectuer des demandes programmatiques de AWS ressources à l'aide de l' AWS API AWS CLI or (à l'aide des AWS kits de développement logiciel). Les informations d'identification temporaires fournissent les mêmes autorisations que les informations d'identification de sécurité à long terme, telles que les informations d'identification de l'utilisateur IAM. Toutefois, il existe quelques différences :

- Lorsque vous passez un appel à l'aide d'informations d'identification de sécurité temporaires, l'appel doit inclure un jeton de session, qui est renvoyé avec ces informations d'identification temporaires. AWS utilise le jeton de session pour valider les informations d'identification de sécurité temporaires.
- Les informations d'identification temporaires arrivent à expiration après un intervalle spécifique. Une fois les informations d'identification temporaires arrivées à expiration, tous les appels que vous effectuez avec elles échoueront. Vous devez donc générer un nouvel ensemble d'informations d'identification temporaires. Les informations d'identification temporaires ne peuvent pas être étendues ou actualisées au-delà de l'intervalle spécifié d'origine.
- Lorsque vous utilisez des informations d'identification temporaires pour effectuer une demande, votre principal peut inclure un ensemble d'étiquettes. Ces balises proviennent de balises de session et de balises attachées au rôle que vous endossez. Pour de plus amples informations sur les balises de session, veuillez consulter Transmettre les balises de session AWS STS.

Si vous utilisez les AWS kits SDK, le AWS Command Line Interface (AWS CLI) ou les Outils pour Windows PowerShell, la manière d'obtenir et d'utiliser des informations d'identification de sécurité temporaires varie selon le contexte. Si vous exécutez du code ou AWS CLI des PowerShell commandes Tools for Windows dans une instance EC2, vous pouvez tirer parti des rôles pour Amazon EC2. Sinon, vous pouvez appeler une API AWS STS pour obtenir les informations d'identification temporaires, puis les utiliser explicitement pour effectuer des appels aux services AWS .

> **Note**
>
> Vous pouvez utiliser AWS Security Token Service (AWS STS) pour créer et fournir à des utilisateurs de confiance des informations d'identification de sécurité temporaires qui peuvent contrôler l'accès à vos AWS ressources. Pour plus d'informations sur AWS STS, consultezInformations d'identification de sécurité temporaires dans IAM. AWS STS est un service mondial dont le point de terminaison par défaut est`https://sts.amazonaws.com`. Ce point de terminaison se trouve dans la région USA Est (Virginie du Nord), bien que les informations d'identification que vous obtenez de ce point de terminaison et d'autres soient valides dans le monde entier. Ces informations d'identification fonctionnent avec les services et les ressources dans n'importe quelle région. Vous pouvez également choisir d'effectuer des appels d' AWS STS API vers des points de terminaison dans l'une des régions prises en charge. Cela permet de réduire la latence en effectuant les demandes depuis les serveurs situés dans une région géographiquement plus proche de vous. Quelle que soit la région d'où proviennent vos informations d'identification, elles fonctionnent dans le monde entier. Pour plus d'informations, veuillez consulter Gérer AWS STS dans un Région AWS.

Table des matières

- Utilisation d'informations d'identification temporaires dans les instances Amazon EC2
- En utilisant des informations d'identification de sécurité temporaires avec AWS Kits SDK
- En utilisant des informations d'identification de sécurité temporaires avec AWS CLI
- Utilisation d'informations d'identification de sécurité temporaires avec des opérations d'API
- En savoir plus

Utilisation d'informations d'identification temporaires dans les instances Amazon EC2

Si vous souhaitez exécuter des AWS CLI commandes ou du code dans une instance EC2, la méthode recommandée pour obtenir des informations d'identification est d'utiliser des rôles pour Amazon EC2. Vous créez un rôle IAM qui spécifie les autorisations que vous souhaitez accorder aux applications qui s'exécutent sur les instances EC2. Lorsque vous lancez l'instance, vous associez le rôle à cette instance.

Les PowerShell commandes Applications et Outils pour Windows qui s'exécutent sur l'instance peuvent ensuite obtenir automatiquement des informations d'identification de sécurité temporaires à partir des métadonnées de l'instance. AWS CLI Vous n'avez pas besoin d'obtenir explicitement

les informations d'identification de sécurité temporaires. Les AWS kits SDK et les AWS CLI outils pour Windows obtiennent PowerShell automatiquement les informations d'identification du service de métadonnées d'instance EC2 (IMDS) et les utilisent. Les informations d'identification temporaires disposent d'autorisations que vous définissez pour le rôle associé à l'instance.

Pour plus d'informations et d'exemples, consultez ce qui suit :

- Utilisation des rôles IAM pour accorder l'accès aux AWS ressources sur Amazon Elastic Compute Cloud — AWS SDK pour Java
- Octroi d'un accès à l'aide d'un rôle IAM — AWS SDK pour .NET
- Création d'un rôle — AWS SDK pour Ruby

En utilisant des informations d'identification de sécurité temporaires avec AWS Kits SDK

Pour utiliser des informations d'identification de sécurité temporaires dans le code, vous appelez par programmation une AWS STS API like `AssumeRole` et vous extrayez les informations d'identification et le jeton de session qui en résultent. Vous utilisez ensuite ces valeurs comme informations d'identification pour les appels ultérieurs à AWS. L'exemple suivant montre un pseudocode expliquant comment utiliser des informations d'identification de sécurité temporaires si vous utilisez un AWS SDK :

```
assumeRoleResult = AssumeRole(role-arn);
tempCredentials = new SessionAWSCredentials(
   assumeRoleResult.AccessKeyId,
   assumeRoleResult.SecretAccessKey,
   assumeRoleResult.SessionToken);
s3Request = CreateAmazonS3Client(tempCredentials);
```

Pour un exemple écrit en Python (en utilisant l'AWS SDK pour Python (Boto)), veuillez consulter Passez à un rôle IAM (AWS API). Cet exemple montre comment appeler `AssumeRole` pour obtenir des informations d'identification de sécurité temporaires, puis utiliser ces informations d'identification pour appeler Amazon S3.

Pour plus d'informations sur la façon d'appeler `AssumeRole`, `GetFederationToken` et d'autres opérations d'API, consultez la Référence sur l'API AWS Security Token Service. Pour plus d'informations sur l'obtention des informations d'identification de sécurité temporaires et du jeton de session à partir des résultats, consultez la documentation du kit SDK que vous utilisez. Vous

trouverez la documentation de tous les AWS SDK sur la page de AWS documentation principale, dans la section SDK et boîtes à outils.

Vous devez vérifier que vous obtenez un nouvel ensemble d'informations d'identification avant que les anciennes arrivent à expiration. Dans certains kits SDK, vous pouvez utiliser un fournisseur qui gère le processus d'actualisation des informations d'identification pour vous. Consultez la documentation du kit SDK que vous utilisez.

En utilisant des informations d'identification de sécurité temporaires avec AWS CLI

Vous pouvez utiliser les informations d'identification de sécurité temporaires avec l'interface AWS CLI. Cela peut vous être utile lors de test de politiques.

À l'aide de l'AWS CLI, vous pouvez appeler une API AWS STS telle que `AssumeRole` ou `GetFederationToken`, puis capturer la sortie obtenue. L'exemple suivant illustre un appel à `AssumeRole` qui envoie la sortie dans un fichier. Dans l'exemple, le `profile` paramètre est supposé être un profil dans le fichier AWS CLI de configuration. Il est également supposé référencer les informations d'identification d'un utilisateur IAM qui a les autorisations d'endosser le rôle.

```
aws sts assume-role --role-arn arn:aws:iam::123456789012:role/role-name --role-session-
name "RoleSession1" --profile IAM-user-name > assume-role-output.txt
```

Une fois l'exécution de la commande terminée, vous pouvez extraire l'ID de clé d'accès, la clé d'accès secrète et le jeton de session de la variable. Vous pouvez le faire manuellement ou à l'aide d'un script. Vous pouvez ensuite affecter ces valeurs aux variables d'environnement.

Lorsque vous exécutez AWS CLI des commandes, elles recherchent les AWS CLI informations d'identification dans un ordre spécifique, d'abord dans les variables d'environnement, puis dans le fichier de configuration. Par conséquent, une fois que vous avez saisi les informations d'identification temporaires dans les variables d'environnement, elles sont AWS CLI utilisées par défaut. (Si vous spécifiez un `profile` paramètre dans la commande, les variables d'environnement AWS CLI sont ignorées. Au lieu de cela AWS CLI , l'analyse se trouve dans le fichier de configuration, ce qui vous permet de remplacer les informations d'identification dans les variables d'environnement si nécessaire.)

L'exemple suivant montre comment définir les variables d'environnement pour les informations d'identification de sécurité temporaires, puis appeler une AWS CLI commande. Comme aucun `profile` paramètre n'est inclus dans la AWS CLI commande, AWS CLI elle recherche d'abord

les informations d'identification dans les variables d'environnement et utilise donc les informations d'identification temporaires.

Linux

```
$ export AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of session token>
$ aws ec2 describe-instances --region us-west-1
```

Windows

```
C:\> SET AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
C:\> SET AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
C:\> SET AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of token>
C:\> aws ec2 describe-instances --region us-west-1
```

Utilisation d'informations d'identification de sécurité temporaires avec des opérations d'API

Si vous envoyez des requêtes d'API HTTPS directes à AWS, vous pouvez signer ces demandes avec les informations d'identification de sécurité temporaires que vous obtenez à partir du AWS Security Token Service (AWS STS). Pour ce faire, vous utilisez l'ID de clé d'accès et la clé d'accès secrète que vous recevez AWS STS. Vous utilisez l'ID de clé d'accès et la clé d'accès secrète de la même façon que vous utiliseriez des informations d'identification à long terme pour signer une demande. Vous ajoutez également à votre demande d'API le jeton de session que vous recevez AWS STS. Vous ajoutez le jeton de session à un en-tête HTTP ou à un paramètre de chaîne de requête appelé `X-Amz-Security-Token`. Vous ajoutez le jeton de session à l'en-tête HTTP ou au paramètre de chaîne de requête, mais pas les deux. Pour plus d'informations sur la signature des demandes d'API HTTPS, consultez la section Signature des demandes d' AWS API dans le Références générales AWS.

En savoir plus

Pour plus d'informations sur l'utilisation AWS STS avec d'autres AWS services, consultez les liens suivants :

- Amazon S3. Veuillez consulter Making Requests Using IAM User Temporary Credentials (Envoyer des demandes au moyen d'informations d'identification temporaires de l'utilisateur IAM) ou Making

Requests Using Federated User Temporary Credentials (Envoyer des demandes au moyen d'informations d'identification temporaires de l'utilisateur fédéré) dans le Guide de l'utilisateur Amazon Simple Storage Service.

- Amazon SNS. Consultez la rubrique Utilisation de politiques basées sur une identité avec Amazon SNS dans le Guide du développeur Amazon Simple Notification Service.
- Amazon SQS. Consultez Identity and Access Management dans Amazon SQS dans le Guide du développeur Amazon Simple Queue Service
- Amazon SimpleDB. Veuillez consulter Using Temporary Security Credentials (Utilisation d'informations d'identification de sécurité temporaires) dans le Manuel du développeur Amazon SimpleDB.

Autorisations affectées aux informations d'identification de sécurité temporaires

Vous pouvez utiliser AWS Security Token Service (AWS STS) pour créer et fournir à des utilisateurs de confiance des informations d'identification de sécurité temporaires qui peuvent contrôler l'accès à vos AWS ressources. Pour plus d'informations sur AWS STS, voirInformations d'identification de sécurité temporaires dans IAM. Lorsqu' AWS STS émet des informations d'identification de sécurité temporaires, celles-ci sont valides jusqu'au délai d'expiration spécifié et elles ne peuvent pas être annulées. Toutefois, les autorisations octroyées aux informations d'identification de sécurité temporaires sont évaluées chaque fois qu'une demande est effectuée à l'aide de ces informations. Il est donc possible de révoquer les informations d'identification en modifiant leurs droits d'accès après leur émission.

Les rubriques suivantes supposent que vous avez une connaissance pratique des AWS autorisations et des politiques. Pour plus d'informations sur ces rubriques, consultez Gestion des accès pour AWS resources.

Rubriques

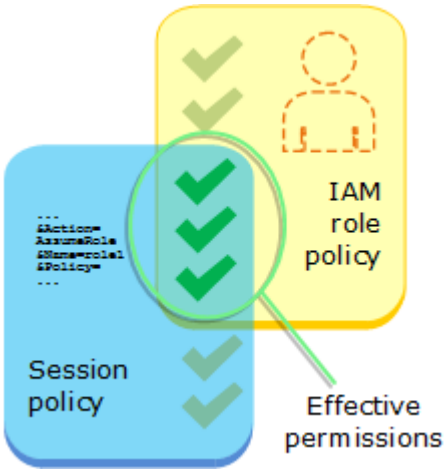
- Autorisations pour AssumeRole AssumeRoleWithSAML, et AssumeRoleWithWebIdentity
- Surveiller et contrôler les actions prises avec les rôles endossés
- Autorisations pour GetFederationToken
- Autorisations pour GetSessionToken
- Désactivation des autorisations affectées aux informations d'identification de sécurité temporaires
- Octroi d'autorisations pour créer des informations d'identification de sécurité temporaires

- Octroi d'autorisations pour l'utilisation de sessions de console à identité renforcée

Autorisations pour AssumeRole AssumeRoleWithSAML, et AssumeRoleWithWebIdentity

La politique d'autorisations du rôle endossé actuellement déterminé les autorisations octroyées aux informations d'identification de sécurité temporaires renvoyées par `AssumeRole`, `AssumeRoleWithSAML` et `AssumeRoleWithWebIdentity`. Vous définissez ces autorisations lors de la définition ou de la mise à jour du rôle.

Le cas échéant, vous pouvez transmettre des politiques de session en ligne ou gérées en tant que paramètre des opérations d'API `AssumeRole`, `AssumeRoleWithSAML` ou `AssumeRoleWithWebIdentity`. Les politiques de session limitent les autorisations pour la session d'informations d'identification temporaires du rôle. Les autorisations de la session obtenues sont une combinaison des politiques basées sur l'identité du rôle et des politiques de session. Vous pouvez utiliser les informations d'identification temporaires du rôle lors des appels d' AWS API suivants pour accéder aux ressources du compte propriétaire du rôle. Vous ne pouvez pas utiliser les politiques de session pour accorder plus d'autorisations que celles autorisées par la politique basée sur l'identité du rôle endossé actuellement. Pour en savoir plus sur la façon dont AWS détermine les autorisations effectives d'un rôle, consultez Logique d'évaluation de politiques.



Les politiques associées aux informations d'identification à l'origine de l'appel ne `AssumeRole` sont pas évaluées AWS lors de la prise de la décision d' « autoriser » ou de « refuser » l'autorisation. L'utilisateur renonce temporairement à ses autorisations d'origine en faveur de celles affectées au rôle endossé. Dans le cas des opérations `AssumeRoleWithSAML` et de l'`AssumeRoleWithWebIdentity`API, il n'y a aucune politique à évaluer car l'appelant de l'API n'est pas une AWS identité.

Exemple : attribution d'autorisations à l'aide de AssumeRole

Vous pouvez utiliser l'opération d'API `AssumeRole` avec différents types de politiques. Voici quelques exemples.

Politique d'autorisations de rôle

Dans cet exemple, vous appelez l'opération d'API `AssumeRole` sans spécifier la politique de session dans le paramètre `Policy` facultatif. Les autorisations affectées aux informations d'identification temporaires sont déterminées par la politique d'autorisations du rôle endossé. L'exemple de politique d'autorisations suivant accorde au rôle l'autorisation de répertorier tous les objets contenus dans un compartiment S3 nommé `productionapp`. Il permet également au rôle d'obtenir, de placer et de supprimer des objets dans ce compartiment.

Example Exemple de politique d'autorisations de rôle

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Politique de session transmise en tant que paramètre

Imaginons que vous souhaitiez autoriser un utilisateur à endosser le même rôle que dans l'exemple suivant. Mais, dans ce cas, vous voulez que la session de rôle ait uniquement l'autorisation d'obtenir et de placer des objets dans le compartiment `productionapp` S3. Vous ne souhaitez pas l'autoriser à supprimer des objets. Pour cela, vous pouvez créer un rôle et spécifier les autorisations souhaitées dans la politique d'autorisations du rôle. Vous pouvez également appeler l'API `AssumeRole` et inclure des politiques de session dans le paramètre `Policy` facultatif dans le cadre de l'opération d'API. Les autorisations de la session obtenues sont une combinaison des politiques basées sur l'identité du rôle et des politiques de session. Les politiques de session ne peuvent pas être utilisées pour accorder plus d'autorisations que celles autorisées par la politique basée sur l'identité du rôle endossé actuellement. Pour de plus amples informations sur les autorisations de session de rôle, veuillez consulter Politiques de session.

Une fois que vous avez récupéré les informations d'identification temporaires de la nouvelle session, vous pouvez les transmettre à l'utilisateur auquel vous voulez donner ces autorisations.

Par exemple, imaginons que la politique suivante soit transmise en tant que paramètre de l'appel d'API. La personne qui utilise la session dispose d'autorisations pour effectuer uniquement les actions suivantes :

- Répertorier tous les objets du compartiment `productionapp`.
- Obtenir et placer des objets dans le compartiment `productionapp`.

Dans la session suivante, l'autorisation `s3:DeleteObject` est exclue du filtre et la session endossée n'obtient pas l'autorisation `s3:DeleteObject`. La politique définit les autorisations maximales pour la session de rôle afin qu'elle remplace les politiques d'autorisations existantes sur le rôle.

Example Exemple de politique de session transmise avec l'appel d'API **`AssumeRole`**

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Resource-based politique

Certaines AWS ressources prennent en charge les politiques basées sur les ressources, et ces politiques fournissent un autre mécanisme pour définir les autorisations qui affectent les informations d'identification de sécurité temporaires. Seules quelques ressources, comme les compartiments Amazon S3, les rubriques Amazon SNS et les files d'attente Amazon SQS, prennent en charge les politiques basées sur des ressources. L'exemple suivant développe les exemples précédents, à l'aide d'un compartiment S3 nommé `productionapp`. La politique suivante est attachée au compartiment.

Lorsque vous attachez la politique basée sur les ressources suivante au compartiment `productionapp`, tous les utilisateurs se voient refuser l'autorisation de supprimer des objets du compartiment. (Voir l'élément `Principal` dans la politique.) Cela inclut tous les utilisateurs du rôle endossé, même si la politique d'autorisations du rôle accorde l'autorisation `DeleteObject`. Une instruction `Deny` explicite a toujours priorité sur une instruction `Allow`.

Example Exemple de politique de compartiment

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Principal": {"AWS": "*"},
    "Effect": "Deny",
    "Action": "s3:DeleteObject",
    "Resource": "arn:aws:s3:::productionapp/*"
```

```
  }
}
```

Pour plus d'informations sur la façon dont plusieurs types de politiques sont combinés et évalués par AWS, voirLogique d'évaluation de politiques.

Surveiller et contrôler les actions prises avec les rôles endossés

Un rôle IAM est un objet dans IAM auquel sont affectées des autorisations. Lorsque vous assumez ce rôle à l'aide d'une identité IAM ou d'une identité externe AWS, vous recevez une session avec les autorisations attribuées au rôle.

Lorsque vous effectuez des actions AWS, les informations relatives à votre session peuvent être enregistrées AWS CloudTrail pour que l'administrateur de votre compte puisse les surveiller. Les administrateurs peuvent configurer les rôles de sorte à exiger des identités qu'elles transmettent une chaîne personnalisée identifiant la personne ou l'application qui effectue des actions dans AWS. Ces informations d'identité sont stockées en tant qu'identité source dans AWS CloudTrail. Lorsque l'administrateur passe en revue l'activité dans CloudTrail, il peut consulter les informations d'identité de la source pour déterminer qui ou quoi a effectué des actions dans le cadre de sessions de rôles supposés.

Une fois qu'une identité de source est définie, elle est présente dans les demandes pour toute AWS action entreprise au cours de la session de rôle. La valeur définie persiste lorsqu'un rôle est utilisé pour assumer un autre rôle via l' AWS API AWS CLI or, ce que l'on appelle le chaînage de rôles. La valeur définie ne peut pas être modifiée durant la session de rôle.

Les administrateurs peuvent configurer des autorisations granulaires en fonction de la présence ou de la valeur de l'identité source afin de mieux contrôler les AWS actions entreprises avec les rôles partagés. Vous pouvez décider si l'attribut d'identité source peut être utilisé, s'il est requis ; et quelle valeur peut être utilisée.

L'utilisation de l'identité source est radicalement différente de celle du nom de session de rôle et des balises de session. La valeur d'identité source ne peut pas être modifiée une fois qu'elle est définie, et elle persiste pour toutes les actions supplémentaires qui sont prises durant la session de rôle. Voici comment utiliser les balises de session et le nom de session de rôle :

- Session tags (Balises de session) : vous pouvez transmettre des balises de session lorsque vous endossez un rôle ou fédérez un utilisateur. Les balises de session sont présentes lorsqu'un rôle

est endossé. Vous pouvez définir des politiques qui utilisent des clés de condition de balise pour accorder des autorisations à vos principaux en fonction de leurs balises. Vous pouvez ensuite consulter CloudTrail les demandes faites pour assumer des rôles ou fédérer des utilisateurs. Pour en savoir plus sur les balises de session, veuillez consulter Transmettre les balises de session AWS STS.

- Role session name (Nom de la session de rôle) : vous pouvez utiliser la clé de condition `sts:RoleSessionName` dans une politique d'approbation de rôle pour exiger que vos utilisateurs fournissent un nom de session spécifique lorsqu'ils endossent un rôle. Le nom de session de rôle peut servir à différencier les sessions de rôle lorsqu'un rôle est utilisé par différents principaux. Pour en savoir plus sur le nom de la session de rôle, voir sts : RoleSessionName.

Nous vous recommandons d'utiliser l'identité source pour contrôler l'identité qui endosse un rôle. L'identité de la source est également utile pour l'exploration CloudTrail des journaux afin de déterminer qui a utilisé le rôle pour effectuer des actions.

Rubriques

- Configuration d'utilisation de l'identité source
- Ce qu'il faut savoir sur l'identité source
- Autorisations requises pour définir l'identité source
- Spécification d'une identité source lorsqu'un rôle est endossé
- Utilisation de l'identité de la source avec AssumeRole
- Utilisation de l'identité de la source avec AssumeRoleWithSAML
- Utilisation de l'identité de la source avec AssumeRoleWithWebIdentity
- Contrôle de l'accès au moyen des informations d'identité source
- Affichage de l'identité de la source dans CloudTrail

Configuration d'utilisation de l'identité source

Votre configuration d'utilisation de l'identité source dépend de la méthode employée lorsque vos rôles sont endossés. Par exemple, vos utilisateurs IAM peuvent endosser des rôles directement via l'opération `AssumeRole`. Si vous possédez des identités d'entreprise, également appelées identités de personnel, elles peuvent accéder à vos AWS ressources en utilisant`AssumeRoleWithSAML`. Si des utilisateurs finaux accèdent à vos applications mobiles ou Web, ils peuvent le faire en utilisant `AssumeRoleWithWebIdentity`. La présentation du flux de haut niveau qui suit vous aidera à

comprendre comment effectuer une configuration pour utiliser les informations d'identité source dans votre environnement existant.

1. Configurer les utilisateurs et les rôles test : en utilisant un environnement de préproduction, configurez les utilisateurs et les rôles test, et configurez leurs politiques afin d'autoriser la définition d'une identité source.

   Si vous utilisez un fournisseur d'identité (IdP) pour vos identités fédérées, configurez-le de sorte à transmettre un attribut utilisateur de votre choix pour l'identité source dans l'assertion ou le jeton.
2. Assumer le rôle : testez le fait d'endosser des rôles et de transmettre une identité source avec les utilisateurs et les rôles que vous configurez pour le test.
3. Révision CloudTrail : vérifiez les informations d'identité de la source pour vos rôles de test dans vos CloudTrail journaux.
4. Entraîner vos utilisateurs : après avoir effectué des tests dans votre environnement de préproduction, vérifiez que vos utilisateurs savent parfaitement comment transmettre les informations d'identité source, si nécessaire. Définissez une date limite à laquelle vous exigerez de vos utilisateurs qu'ils fournissent une identité source dans votre environnement de production.
5. Configurer des politiques de production : configurez des politiques pour votre environnement de production, puis ajoutez-les à vos utilisateurs et rôles de production.
6. Surveillez l'activité : surveillez l'activité de votre rôle de production à l'aide de CloudTrail journaux.

Ce qu'il faut savoir sur l'identité source

Gardez ce qui suit à l'esprit lorsque vous travaillez avec l'identité source.

- Les politiques d'approbation pour tous les rôles connectés à un fournisseur d'identité doivent disposer de l'autorisation `sts:SetSourceIdentity`. Pour les rôles qui ne disposent pas de cette autorisation dans la politique d'approbation de rôle, l'opération `AssumeRole*` échouera. Si vous ne souhaitez pas mettre à jour la politique d'approbation de rôle pour chaque rôle, vous pouvez utiliser une instance de fournisseur d'identité distincte pour transmettre l'identité source. Ajoutez ensuite l'autorisation `sts:SetSourceIdentity` uniquement aux rôles qui sont connectés au fournisseur d'identité distinct.
- Lorsqu'une identité définit une identité source, la clé `sts:SourceIdentity` est présente dans la demande. Pour les actions qui seront prises ultérieurement durant la session de rôle, la clé `aws:SourceIdentity` est présente dans la demande. AWS ne contrôle la valeur de l'identité source dans aucune des clés `sts:SourceIdentity` ou `aws:SourceIdentity`. Si vous

choisissez d'exiger une identité source, vous devez choisir un attribut que vos utilisateurs ou votre IdP doivent fournir. Pour des raisons de sécurité, vous devez vérifier votre capacité à contrôler la façon dont ces valeurs sont fournies.

- La valeur de l'identité de la source doit comporter entre 2 et 256 caractères et ne peut contenir que des caractères alphanumériques, des traits de soulignement et les caractères suivants :, + = @ - (tiret). Vous ne pouvez pas utiliser une valeur qui commence par le texte **`aws:`**. Ce préfixe est réservé à un usage AWS interne.
- Les informations d'identité de la source ne sont pas capturées CloudTrail lorsqu'un AWS service ou un rôle lié à un service effectue une action pour le compte d'une identité fédérée ou d'un personnel.

> **Important**
>
> Vous ne pouvez pas passer à un rôle dans le Console de gestion AWS qui nécessite la définition d'une identité de source lorsque le rôle est assumé. Pour assumer ce rôle, vous pouvez utiliser l' AWS API AWS CLI or pour appeler l'`AssumeRole`opération et spécifier le paramètre d'identité de la source.

Autorisations requises pour définir l'identité source

En plus de l'action qui correspond à l'opération d'API, vous devez disposer de l'action d'autorisation suivante dans votre politique :

```
sts:SetSourceIdentity
```

- Pour spécifier une identité source, les principaux (utilisateurs et rôles IAM) doivent disposer des autorisations nécessaires pour `sts:SetSourceIdentity`. Votre qualité d'administrateur vous permet de configurer cela dans la politique de confiance de rôle et la politique d'autorisations du principal.
- Lorsque vous endossez un rôle avec un autre rôle (chaînage de rôles), des autorisations sont exigées pour `sts:SetSourceIdentity` tant dans la politique d'autorisations du principal qui endosse le rôle que dans la politique de confiance de rôle du rôle cible. Sinon, l'opération consistant à endosser le rôle échouera.
- Lorsque vous utilisez l'identité source, les politiques d'approbation de rôle pour tous les rôles connectés à un fournisseur d'identité doivent disposer de l'autorisation `sts:SetSourceIdentity`. L'opération `AssumeRole*` échouera pour tout rôle connecté à un

fournisseur d'identité sans cette autorisation. Si vous ne voulez pas mettre à jour la politique de confiance de rôle pour chaque rôle, vous pouvez utiliser une instance IdP distincte pour transmettre l'identité source et ajouter l'autorisation `sts:SetSourceIdentity` aux seuls rôles qui sont connectés à l'instance IdP distincte.

- Pour définir une identité source au-delà des limites de compte, vous devez inclure l'autorisation `sts:SetSourceIdentity` à deux endroits. Elle doit être présente dans la politique d'autorisations du principal dans le compte d'origine et dans la politique de confiance de rôle du rôle dans le compte cible. Vous devrez peut-être faire cela lorsqu'un rôle est utilisé pour endosser un rôle dans un autre compte (chaînage de rôles).

Supposons, par exemple, qu'en tant qu'administrateur de compte, vous vouliez autoriser l'utilisateur IAM `DevUser` dans votre compte à endosser le `Developer_Role`dans le même compte. Mais vous voulez autoriser cette action uniquement si l'utilisateur a défini l'identité source à son nom d'utilisateur IAM. Vous pouvez attacher la politique suivante à l'utilisateur IAM.

Example Exemple de politique basée sur l'identité associée à DevUser

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeRole",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role"
    },
    {
      "Sid": "SetAwsUserNameAsSourceIdentity",
      "Effect": "Allow",
      "Action": "sts:SetSourceIdentity",
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role",
      "Condition": {
        "StringLike": {
          "sts:SourceIdentity": "${aws:username}"
        }
      }
    }
  ]
```

```
}
```

Pour appliquer les valeurs d'identité source acceptables, vous pouvez configurer la politique de confiance de rôle suivante. La politique donne à l'utilisateur IAM les autorisations `DevUser` pour endosser le rôle et définir une identité source. La clé de condition `sts:SourceIdentity` définit la valeur d'identité source acceptable.

Example Exemple de politique de confiance de rôle pour une identité source

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDevUserAssumeRole",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/DevUser"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "sts:SourceIdentity": "DevUser"
        }
      }
    }
  ]
}
```

À l'aide des informations d'identification de l'utilisateur IAM`DevUser`, celui-ci tente de supposer qu'il `DeveloperRole` utilise la AWS CLI requête suivante.

Example Exemple de requête AssumeRole CLI

```
aws sts assume-role \
```

```
--role-arn arn:aws:iam::123456789012:role/Developer_Role \
--role-session-name Dev-project \
--source-identity DevUser \
```

Lors de l' AWS évaluation de la demande, le contexte de la demande contient le signe `sts:SourceIdentity` de`DevUser`.

Spécification d'une identité source lorsqu'un rôle est endossé

Vous pouvez spécifier une identité de source lorsque vous utilisez l'une des opérations d' AWS STS `AssumeRole*`API pour obtenir des informations d'identification de sécurité temporaires pour un rôle. L'opération d'API que vous utilisez varie selon votre cas d'utilisation.

Par exemple, si vous utilisez des rôles IAM pour permettre aux utilisateurs IAM d'accéder à AWS des ressources auxquelles ils n'ont pas normalement accès, vous pouvez utiliser cette opération. `AssumeRole` Si vous utilisez la fédération d'identités d'entreprise pour gérer vos utilisateurs de main-d'œuvre, vous pouvez utiliser l'opération `AssumeRoleWithSAML`. Si vous utilisez la fédération OIDC pour autoriser des utilisateurs finaux à accéder à vos applications mobiles ou Web, vous pouvez utiliser l'opération `AssumeRoleWithWebIdentity`. Les sections suivantes détaillent l'utilisation de l'identité source avec chaque opération. Pour en savoir plus sur les scénarios courants d'informations d'identification temporaires, veuillez consulter Scénarios courants d'informations d'identification temporaires.

Utilisation de l'identité de la source avec AssumeRole

L'`AssumeRole`opération renvoie un ensemble d'informations d'identification temporaires que vous pouvez utiliser pour accéder aux AWS ressources. Vous pouvez utiliser l'utilisateur ou les informations d'identification du rôle IAM pour appeler `AssumeRole`. Pour transmettre l'identité de la source tout en assumant un rôle, utilisez l'`--source-identity` AWS CLI option ou le paramètre `SourceIdentity` AWS d'API. L'exemple suivant vous explique comment spécifier l'identité source avec la AWS CLI.

Example Exemple de requête AssumeRole CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/developer \
--role-session-name Audit \
--source-identity Admin \
```

Utilisation de l'identité de la source avec AssumeRoleWithSAML

Le principal qui appelle l'`AssumeRoleWithSAML`opération est authentifié à l'aide de SAML-based la fédération. Cette opération renvoie un ensemble d'informations d'identification temporaires que vous pouvez utiliser pour accéder aux AWS ressources. Pour plus d'informations sur l'utilisation SAML-based de la fédération pour Console de gestion AWS l'accès, consultezPermettre aux principaux fédérés SAML 2.0 d'accéder au Console de gestion AWS. Pour plus de détails sur AWS CLI notre accès à l' AWS API, consultezFédération SAML 2.0. Pour un didacticiel sur la configuration de la fédération SAML pour vos utilisateurs Active Directory, consultez la section Authentification AWS fédérée avec Active Directory Federation Services (ADFS) dans le AWS blog de sécurité.

En tant qu'administrateur, vous pouvez autoriser les membres de l'annuaire de votre entreprise à se fédérer pour AWS utiliser l' AWS STS `AssumeRoleWithSAML`opération. Pour ce faire, effectuez les tâches suivantes :

1. Configurer un fournisseur SAML dans votre organisation.
2. Créer un fournisseur SAML dans IAM
3. Configurez un rôle et ses autorisations AWS pour vos principaux fédérés SAML.
4. Achever la configuration de l'IdP SAML et créer des assertions pour la réponse d'authentification SAML.

Pour définir un attribut SAML pour l'identité source, incluez dans l'élément `Attribute` l'attribut `Name` défini à l'adresse `https://aws.amazon.com/SAML/Attributes/SourceIdentity`. Utilisez l'élément `AttributeValue` pour spécifier la valeur de l'identité source. Par exemple, supposons que vous souhaitez transmettre l'attribut d'identité suivant en tant qu'identité source :

`SourceIdentity:DiegoRamirez`

Pour transmettre cet attribut, incluez l'élément suivant dans votre assertion SAML.

Example Extrait d'une assertion SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
<AttributeValue>DiegoRamirez</AttributeValue>
</Attribute>
```

Utilisation de l'identité de la source avec AssumeRoleWithWebIdentity

Le principal qui appelle l'opération `AssumeRoleWithWebIdentity` est authentifié à l'aide de la fédération compatible OpenID Connect (OIDC). Cette opération renvoie un ensemble d'informations d'identification temporaires que vous pouvez utiliser pour accéder aux ressources AWS . Pour de plus amples informations sur l'utilisation de la fédération OIDC pour l'accès à l' Console de gestion AWS , consultez Fédération OIDC.

Pour transmettre l'identité source depuis OpenID Connect (OIDC), vous devez inclure l'identité source dans le jeton web JSON (JWT). Incluez l'identité source dans l'espace de noms `https://aws.amazon.com/` `source_identity` dans le jeton lorsque vous soumettez la demande `AssumeRoleWithWebIdentity`. Pour en savoir plus sur les jetons OIDC et les revendications, veuillez consulter Utilisation des jetons avec les groupes d'utilisateurs dans le Guide du développeur Amazon Cognito .

Par exemple, le JWT décodé suivant est un jeton utilisé pour appeler `AssumeRoleWithWebIdentity` avec l'identité source `Admin`.

Example Exemple de jeton web JSON décodé

```
{
    "sub": "john",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/source_identity":"Admin"
}
```

Contrôle de l'accès au moyen des informations d'identité source

Lorsqu'une identité de source est initialement définie, la SourceIdentity clé sts : est présente dans la demande. Une fois l'identité de source définie, la SourceIdentity clé aws : est présente dans toutes les demandes suivantes effectuées au cours de la session de rôle. En tant qu'administrateur, vous pouvez rédiger des politiques qui accordent une autorisation conditionnelle pour effectuer des AWS actions en fonction de l'existence ou de la valeur de l'attribut d'identité source.

Imaginez que vous souhaitiez demander à vos développeurs de définir une identité de source pour qu'ils assument un rôle essentiel et soient autorisés à écrire sur une AWS ressource critique pour la

production. Imaginez également que vous accordiez AWS l'accès aux identités de vos employés en utilisant`AssumeRoleWithSAML`. Comme vous voulez que seuls les développeurs senior Saanvi et Diego aient accès au rôle, vous créez a politique de confiance suivante pour le rôle.

Example Exemple de politique d'approbation de rôle pour une identité source (SAML)

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "SAMLProviderAssumeRoleWithSAML",
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:AssumeRoleWithSAML"
      ],
      "Condition": {
        "StringEquals": {
          "SAML:aud": "https://region-code.signin.aws.amazon.com/saml"
        }
      }
    },
    {
      "Sid": "SetSourceIdentitySrEngs",
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringLike": {
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
          ]
```

```
        }
      }
    }
  ]
}
```

La politique d'approbation contient une condition pour l'interface `sts:SourceIdentity` qui exige une identité source de Saanvi ou Diego pour endosser le rôle critique.

Par ailleurs, si vous utilisez un fournisseur OIDC pour la fédération et que les utilisateurs sont authentifiés avec l'interface `AssumeRoleWithWebIdentity`, votre politique d'approbation de rôle peut se présenter comme suit.

Example Exemple de politique d'approbation de rôle pour l'identité source (fournisseur OIDC)

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/server.example.com"
      },
      "Action": [
        "sts:AssumeRoleWithWebIdentity",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "server.example.com:aud": "oidc-audience-id"
        },
        "StringLike": {
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
          ]
        }
      }
    }
```

```
  ]
}
```

Chaînage de rôles et exigences entre comptes

Supposons que vous vouliez autoriser les utilisateurs ayant endossé un `CriticalRole` à endosser un `CriticalRole_2` dans un autre compte. Les informations d'identification de session de rôle qui ont été obtenues pour endosser le `CriticalRole` sont utilisées pour effectuer un chaînage de rôles à un second rôle, `CriticalRole_2`, dans un autre compte. Le rôle est endossé au-delà d'une limite de compte. Par conséquent, l'autorisation `sts:SetSourceIdentity` doit être octroyée tant dans la politique d'autorisations sur le `CriticalRole` que dans la politique de confiance de rôle sur le `CriticalRole_2`.

Example Exemple de politique d'autorisations sur CriticalRole

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeRoleAndSetSourceIdentity",
      "Effect": "Allow",
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Resource": "arn:aws:iam::222222222222:role/CriticalRole_2"
    }
  ]
}
```

Afin de sécuriser la définition de l'identité source au-delà de la limite du compte, la politique de confiance de rôle suivante fait uniquement confiance au principal de rôle pour `CriticalRole` pour définir l'identité source.

Example Exemple de politique de confiance des rôles sur CriticalRole _2

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111111111111:role/CriticalRole"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringLike": {
          "aws:SourceIdentity": ["Saanvi","Diego"]
        }
      }
    }
  ]
}
```

L'utilisateur effectue l'appel suivant à l'aide des informations d'identification de session de rôle obtenues en supposant CriticalRole. L'identité de la source a été définie lors de l'hypothèse de CriticalRole, il n'est donc pas nécessaire de la définir à nouveau explicitement. Si l'utilisateur tente de définir une identité source différente de la valeur définie lors de la supposition de `CriticalRole`, la demande d'endosser le rôle sera refusée.

Example Exemple de requête AssumeRole CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::222222222222:role/CriticalRole_2 \
--role-session-name Audit \
```

Lorsque le principal appelant endosse le rôle, l'identité source dans la demande persiste à partir de la première session de rôle endossée. Par conséquent, les clés `aws:SourceIdentity` et `sts:SourceIdentity` sont toutes deux présentes dans le contexte de demande.

Affichage de l'identité de la source dans CloudTrail

Vous pouvez l'utiliser CloudTrail pour afficher les demandes faites pour assumer des rôles ou fédérer des utilisateurs. Vous pouvez également afficher les demandes de rôle ou d'utilisateur souhaitant effectuer des actions dans AWS. Le fichier CloudTrail journal contient des informations sur l'identité de source définie pour la session utilisateur fédérée ou avec rôle assumé. Pour de plus amples informations, consultez Journalisation IAM et AWS STS Appels d'API avec AWS CloudTrail.

Supposons, par exemple, qu'un utilisateur fasse une AWS STS `AssumeRole` demande et définisse une identité de source. Vous trouverez les `sourceIdentity` informations contenues dans la `requestParameters` clé de votre CloudTrail journal.

Example Exemple de section RequestParameters dans un AWS CloudTrail journal

```
"eventVersion": "1.05",
    "userIdentity": {
        "type": "AWSAccount",
        "principalId": "AIDAJ45Q7YFFAREXAMPLE",
        "accountId": "111122223333"
    },
    "eventTime": "2020-04-02T18:20:53Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.64",
    "userAgent": "aws-cli/1.16.96 Python/3.6.0 Windows/10 botocore/1.12.86",
    "requestParameters": {
        "roleArn": "arn:aws:iam::123456789012:role/DevRole",
        "roleSessionName": "Dev1",
        "sourceIdentity": "source-identity-value-set"
    }
```

Si l'utilisateur utilise la session de rôle supposée pour effectuer une action, les informations d'identité de la source sont présentes dans la `userIdentity` clé du CloudTrail journal.

Example Exemple de clé UserIdentity dans un AWS CloudTrail journal

```
{
```

```
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAJ45Q7YFFAREXAMPLE:Dev1",
    "arn": "arn:aws:sts::123456789012:assumed-role/DevRole/Dev1",
    "accountId": "123456789012",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAJ45Q7YFFAREXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/DevRole",
        "accountId": "123456789012",
        "userName": "DevRole"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-02-21T23:46:28Z"
      },
      "sourceIdentity": "source-identity-value-present"
    }
  }
}
```


Pour voir des exemples AWS STS d'événements d'API dans CloudTrail les journaux, consultezExemples d'événements d'API IAM dans le journal CloudTrail. Pour plus de détails sur les informations contenues dans les fichiers CloudTrail journaux, consultez la section Référence des CloudTrail événements dans le Guide de AWS CloudTrail l'utilisateur.

Autorisations pour GetFederationToken

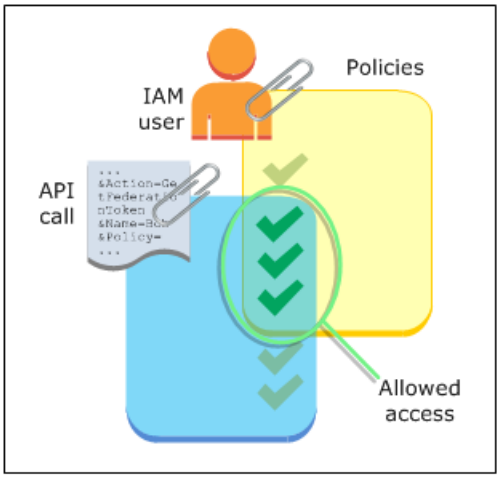
L'opération `GetFederationToken` est appelée par un utilisateur IAM et renvoie les informations d'identification temporaires pour cet utilisateur. Cette opération fédère l'utilisateur. Les autorisations attribuées à une session utilisateur AWS STS fédérée sont définies à l'un des deux endroits suivants :

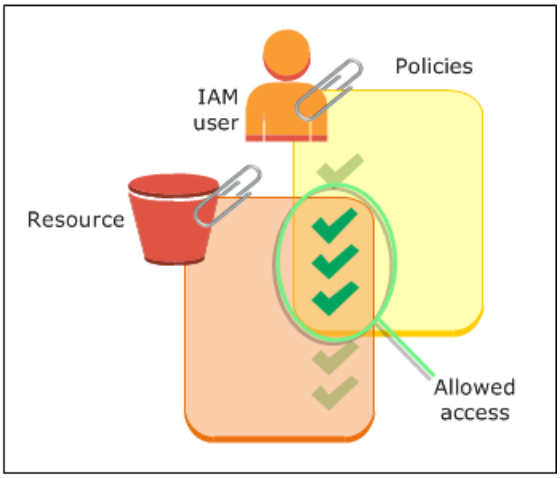
- Les politiques de session transmises en tant que paramètre de l'appel d'API `GetFederationToken`. (Il s'agit du scénario le plus courant.)
- Stratégie basée sur les ressources qui nomme explicitement la session utilisateur AWS STS fédérée dans l'`Principal`élément de la stratégie. (Ce scénario est moins courant.)

Les politiques de session sont des politiques avancées que vous transmettez en tant que paramètres lorsque vous créez par programmation une session temporaire. Lorsque vous créez une session utilisateur AWS STS fédérée et que vous adoptez des politiques de session, les autorisations de session qui en résultent sont à l'intersection de la politique basée sur l'identité de l'utilisateur et des politiques de session. Vous ne pouvez pas utiliser la politique de session pour accorder plus d'autorisations que celles autorisées par la politique basée sur l'identité de l'utilisateur fédéré.

Cela signifie que dans la plupart des cas, si vous ne transmettez pas de politique avec l'appel d'API `GetFederationToken`, les informations d'identification de sécurité temporaires générées ne disposent d'aucune autorisation. Toutefois, une politique basée sur les ressources peut fournir des autorisations supplémentaires pour la session. Vous pouvez accéder à une ressource avec une politique basée sur les ressources, qui spécifie votre session en tant que principal autorisé.

Les illustrations suivantes sont une représentation visuelle de la façon dont les politiques interagissent pour déterminer les autorisations accordées aux informations d'identification de sécurité temporaires retournées par un appel à `GetFederationToken`.





Exemple : attribution d'autorisations à l'aide de GetFederationToken

Vous pouvez utiliser l'action d'API `GetFederationToken` avec différents types de politiques. Voici quelques exemples.

Politique attachée à l'utilisateur IAM

Dans cet exemple, une application cliente basée sur un navigateur s'appuie sur deux services web backend. L'un des backend est votre propre serveur d'authentification qui utilise votre système d'identité pour authentifier l'application cliente. L'autre backend est un service AWS qui fournit certaines des fonctionnalités de l'application cliente. L'application cliente est authentifiée par votre serveur, puis ce dernier créée ou récupère la politique d'autorisation appropriée. Ensuite,

votre serveur appelle l'API `GetFederationToken` afin d'obtenir des informations d'identification de sécurité temporaires, puis il retourne ces informations d'identification à l'application cliente. L'application cliente peut ensuite adresser des demandes directement au AWS service avec les informations d'identification de sécurité temporaires. Cette architecture permet à l'application cliente de faire des AWS demandes sans intégrer d'informations d' AWS identification à long terme.

Votre serveur d'authentification appelle l'API `GetFederationToken` avec les informations d'identification de sécurité à long terme d'un utilisateur IAM nommé `token-app`. Cependant, les informations d'identification de l'utilisateur IAM à long terme restent sur votre serveur et ne sont jamais distribuées au client. L'exemple de politique suivant est attaché à l'utilisateur `token-app` IAM et définit l'ensemble le plus large d'autorisations dont vos utilisateurs AWS STS fédérés (clients) auront besoin. Notez que l'`sts:GetFederationToken`autorisation est requise pour que votre service d'authentification obtienne des informations d'identification de sécurité temporaires pour les utilisateurs AWS STS fédérés.

Example Exemple de politique attachée à l'application **`Token-App`** utilisateur IAM qui appelle **`GetFederationToken`**

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:GetFederationToken",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "dynamodb:ListTables",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sqs:ReceiveMessage",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
```

```
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sns:ListSubscriptions",
      "Resource": "*"
    }
  ]
}
```

La politique précédente accorde plusieurs autorisations à l'utilisateur IAM. Cependant, cette politique à elle seule n'accorde aucune autorisation à l'utilisateur AWS STS fédéré. Si cet utilisateur IAM appelle `GetFederationToken` et ne transmet pas de politique en tant que paramètre de l'appel d'API, l'utilisateur AWS STS fédéré qui en résulte ne dispose d'aucune autorisation effective.

Politique de session transmise en tant que paramètre

La méthode la plus courante pour s'assurer que l'utilisateur AWS STS fédéré dispose des autorisations appropriées consiste à transmettre les politiques de session dans l'appel `GetFederationToken` d'API. En reprenant l'exemple précédent, imaginons que `GetFederationToken` est appelé avec les informations d'identification de l'utilisateur `token-app` IAM. Imaginons ensuite que la politique de session suivante est transmise en tant que paramètre de l'appel d'API. L'utilisateur fédéré AWS STS a obtenu l'autorisation de répertorier le contenu du compartiment Amazon S3 nommé `productionapp`. L'utilisateur ne peut pas effectuer les actions Amazon S3 `GetObject` `PutObject` et `DeleteObject` sur les éléments dans le compartiment `productionapp`.

L'utilisateur fédéré se voit attribuer ces autorisations, car elles sont une combinaison des politiques d'utilisateur IAM et les politiques de session que vous transmettez.

L'utilisateur AWS STS fédéré n'a pas pu effectuer d'actions dans Amazon SNS, Amazon SQS, Amazon DynamoDB ou dans aucun compartiment S3 à l'exception de. `productionapp` Ces actions sont refusées, même si ces autorisations sont accordées à l'utilisateur IAM associé à l'appel `GetFederationToken`.

Example Exemple de politique transmise en tant que paramètre de l'appel d'API **`GetFederationToken`**

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::productionapp"]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": ["arn:aws:s3:::productionapp/*"]
    }
  ]
}
```

Resource-based politiques

Certaines AWS ressources prennent en charge les politiques basées sur les ressources, et ces politiques fournissent un autre mécanisme permettant d'accorder des autorisations directement à un utilisateur AWS STS fédéré. Seuls certains AWS services prennent en charge les politiques basées sur les ressources. Par exemple, Amazon S3 comprend des compartiments, Amazon SNS des rubriques, et Amazon SQS des files d'attente auxquelles vous pouvez attacher des politiques. Pour obtenir la liste de tous les services qui prennent en charge les politiques basées sur les ressources, consultez AWS services qui fonctionnent avec IAM la colonne « Resource-based politiques » des tableaux. Vous pouvez utiliser des politiques basées sur les ressources pour attribuer des autorisations directement à un utilisateur AWS STS fédéré. Pour ce faire, spécifiez l'Amazon Resource Name (ARN) de l'utilisateur AWS STS fédéré dans l'`Principal`élément de la politique basée sur les ressources. L'exemple suivant illustre cette méthode et développe les exemples précédents, à l'aide d'un compartiment S3 nommé `productionapp`.

La politique basée sur les ressources suivante est attachée au compartiment. Cette politique de compartiment permet à un utilisateur AWS STS fédéré nommé Carol d'accéder au compartiment. Lorsque l'exemple de politique décrit précédemment est attaché à l'utilisateur `token-app` IAM, l'utilisateur AWS STS fédéré nommé Carol est autorisé à effectuer les `s3:DeleteObject` actions `s3:GetObjects3:PutObject`, et sur le bucket nommé. `productionapp` Cela est vrai même lorsqu'aucune politique de session n'est transmise en tant que paramètre de l'appel d'API `GetFederationToken`. En effet, dans ce cas, l'utilisateur AWS STS fédéré nommé Carol a reçu des autorisations explicites en vertu de la politique basée sur les ressources suivante.

N'oubliez pas qu'un utilisateur AWS STS fédéré ne reçoit des autorisations que lorsque ces autorisations sont explicitement accordées à la fois à l'utilisateur IAM et à l'utilisateur AWS STS fédéré. Ils peuvent également être accordés (au sein du compte) par une politique basée sur les ressources qui nomme explicitement l'utilisateur AWS STS fédéré dans l'`Principal`élément de la politique, comme dans l'exemple suivant.

Example Exemple de politique de compartiment qui autorise l'accès à l'utilisateur fédéré

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Principal": {
            "AWS": "arn:aws:sts::111122223333:federated-user/Carol"
        },
        "Effect": "Allow",
        "Action": [
            "s3:GetObject",
            "s3:PutObject",
            "s3:DeleteObject"
        ],
        "Resource": [
            "arn:aws:s3:::productionapp/*"
        ]
    }
}
```

Pour plus d'informations sur la manière dont les politiques sont évaluées, consultez la rubrique Logique d'évaluation des politiques.

Autorisations pour GetSessionToken

La bonne occasion pour appeler l'`GetSessionToken` opération d'API ou la `get-session-token` commande CLI est lorsqu'un utilisateur doit être authentifié avec l'authentification multi-facteur (MFA). Il est possible d'écrire une politique qui autorise certaines actions uniquement lorsque ces actions sont requises par un utilisateur authentifié à l'aide de MFA. Pour réussir à transmettre le contrôle de l'autorisation d'authentification MFA, un utilisateur doit d'abord appeler `GetSessionToken` et inclure le `SerialNumber` facultatif et `TokenCode` les paramètres. Si l'utilisateur est correctement authentifié par un dispositif MFA, les informations d'identification retournées par l'opération d'API `GetSessionToken` incluent le contexte MFA. Ce contexte indique que l'utilisateur est authentifié par l'authentification MFA et est autorisé pour les opérations d'API nécessitant une authentification MFA.

Autorisations requises pour GetSessionToken

Aucune autorisation n'est requise pour qu'un utilisateur obtienne un jeton de session. Le but principal de l'opération `GetSessionToken` est d'authentifier l'utilisateur à l'aide de l'authentification MFA. Vous ne pouvez pas utiliser de stratégies pour contrôler les opérations d'authentification.

Pour autoriser l'exécution de la plupart AWS des opérations, vous devez ajouter l'action portant le même nom à une politique. Par exemple, pour créer un utilisateur, vous devez utiliser l'opération d'API `CreateUser`, la commande CLI `create-user` ou l' Console de gestion AWS. Pour effectuer ces opérations, vous devez disposer d'une politique qui vous permet d'accéder à l'action `CreateUser` .

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateUser",
            "Resource": "*"
        }
    ]
}
```

Vous pouvez inclure l'action `GetSessionToken` dans vos politiques, mais elle n'a aucun effet sur la possibilité de l'utilisateur d'effectuer l'opération `GetSessionToken` .

Autorisations accordées par GetSessionToken

Si `GetSessionToken` est appelée avec les informations d'identification d'un utilisateur IAM, les informations d'identification de sécurité temporaires ont les mêmes autorisations que cet utilisateur IAM. De même, `GetSessionToken` s'ils sont appelés avec des Utilisateur racine d'un compte AWS informations d'identification, les informations d'identification de sécurité temporaires ont des autorisations d'utilisateur root.

> **Note**
>
> Il est recommandé de ne pas appeler `GetSessionToken` avec les informations d'identification d'utilisateur racine. Mettez plutôt en application nos bonnes pratiques pour créer des utilisateurs IAM avec les autorisations nécessaires. Utilisez ensuite ces utilisateurs IAM pour vos interactions quotidiennes avec AWS.

Les informations d'identification temporaires que vous obtenez lorsque vous appelez `GetSessionToken` présentent les fonctionnalités et limitations suivantes :

- Vous pouvez utiliser les informations d'identification pour accéder au Console de gestion AWS en les transmettant au point de terminaison d'authentification unique de la fédération à l'`https://signin.aws.amazon.com/federation`adresse. Pour de plus amples informations, veuillez consulter Activez l'accès personnalisé du courtier d'identité au AWS console.
- Vous ne pouvez pas utiliser les informations d'identification pour appeler les opérations IAM ou d'API AWS STS . Vous pouvez les utiliser pour appeler des opérations d'API pour d'autres AWS services.

Comparez cette opération d'API et ses limitations et capacités avec les autres opérations d'API qui créent des informations d'identification de sécurité temporaires à Comparez les AWS STS informations d'identification

Pour plus d'informations sur l'accès aux API protégé par MFA à l'aide de `GetSessionToken`, consultez Accès sécurisé aux API avec MFA.

Désactivation des autorisations affectées aux informations d'identification de sécurité temporaires

Les informations d'identification de sécurité temporaires sont valides jusqu'à la fin de leur délai d'expiration. Ces informations d'identification sont valides pendant la durée spécifiée, de 900 secondes (15 minutes) à un maximum de 129 600 secondes (36 heures). La durée de session par défaut est de 43 200 secondes (12 heures). Vous pouvez révoquer ces informations d'identification, mais vous devez également modifier les autorisations de l'utilisateur ou du rôle IAM afin d'empêcher l'utilisation d'informations d'identification compromises pour des activités de compte malveillantes. Les autorisations attribuées aux informations d'identification de sécurité temporaires sont évaluées chaque fois qu'elles sont utilisées pour faire une AWS demande. Une fois que vous avez supprimé toutes les autorisations des informations d'identification, les AWS demandes qui les utilisent échouent.

Les mises à jour des politiques peuvent prendre quelques minutes pour être mises en vigueur. Pour les sessions de rôle IAM, vous pouvez révoquer les informations d'identification de sécurité temporaires du rôle afin d'obliger tous les utilisateurs assumant le rôle à s'authentifier à nouveau et à demander de nouvelles informations d'identification. Pour plus d'informations, consultez Révoquer les informations d'identification temporaires du rôle.

Vous ne pouvez pas modifier les autorisations pour un Utilisateur racine d'un compte AWS. De même, vous ne pouvez pas modifier les autorisations des informations d'identification de sécurité temporaires créées en appelant `GetFederationToken` ou `GetSessionToken` lorsque vous êtes connecté comme utilisateur racine. C'est pour cette raison que nous vous recommandons de ne pas appeler `GetFederationToken` ou `GetSessionToken` en tant qu'utilisateur racine.

Pour les procédures relatives à la modification des autorisations d'un utilisateur IAM, consultez Modification des autorisations pour un utilisateur IAM.

Pour les procédures relatives à la modification des autorisations d'un rôle IAM, consultez Mettre à jour les autorisations pour un rôle.

> **Important**
>
> Vous ne pouvez pas modifier les rôles dans IAM qui ont été créés à partir des ensembles d'autorisations d'IAM Identity Center. Vous devez révoquer la session d'ensemble d'autorisations active pour un utilisateur dans IAM Identity Center. Pour plus d'informations,

consultez Révocation des sessions de rôle IAM actives créées par des ensembles d'autorisations dans le Guide de l'utilisateur IAM Identity Center.

Rubriques

- Refus d'accès à toutes les sessions de rôle IAM associées à un rôle
- Refus d'accès à une session de rôle IAM spécifique
- Refus d'accès aux sessions utilisant des informations d'identification de sécurité temporaires avec des clés de contexte de condition
- Refus d'accès à un principal spécifique avec des politiques basées sur les ressources

Refus d'accès à toutes les sessions de rôle IAM associées à un rôle

Cette procédure refuse les autorisations à toutes les sessions de rôle IAM associées à un rôle. Utilisez cette méthode lorsque vous craignez un accès suspect des :

- Principaux d'un autre compte utilisant l'accès intercompte
- Identités des utilisateurs externes autorisés à accéder aux AWS ressources de votre compte
- Utilisateurs ayant été authentifiés dans une application mobile ou Web à l'aide d'un fournisseur OIDC

Pour modifier ou supprimer les autorisations affectées aux informations d'identification de sécurité temporaires obtenues en appelant `AssumeRole`, `AssumeRoleWithSAML`, `AssumeRoleWithWebIdentity`, `GetFederationToken` ou `GetSessionToken`, vous pouvez modifier ou supprimer la politique basée sur l'identité qui définit les autorisations pour le rôle.

Important

S'il existe une politique basée sur les ressources qui autorise l'accès au principal, vous devez également ajouter un refus explicite pour cette ressource. Consultez Refus d'accès à un principal spécifique avec des politiques basées sur les ressources pour plus de détails.

Pour refuser l'accès à toutes les sessions de rôle IAM associées à un rôle

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la.

2. Dans le panneau de navigation, choisissez Rôles.
3. Choisissez le nom du rôle à modifier. Vous pouvez utiliser la zone de recherche pour filtrer la liste.
4. Sélectionnez l'onglet Autorisations.
5. Sélectionnez la politique concernée à modifier. Avant de modifier une politique gérée par le client, consultez l'onglet Entités attachées pour éviter de perturber l'accès à d'autres identités auxquelles la même politique est attachée.
6. Choisissez l'onglet JSON et mettez à jour la politique pour refuser toutes les ressources et actions.

> **Note**
>
> Ces autorisations sont les mêmes que celles de la politique AWS gérée AWSDenyAll. Vous pouvez associer cette politique AWS gérée à n'importe quel utilisateur ou rôle IAM auquel vous souhaitez refuser tout accès.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "DenyAll",
            "Effect": "Deny",
            "Action": [
                "*"
            ],
            "Resource": "*"
        }
    ]
}
```

7. Sur la page Vérification, vérifiez le Récapitulatif de la politique, puis choisissez Enregistrer les modifications pour enregistrer votre travail.

Lorsque vous mettez à jour la politique, les modifications affectent les autorisations de toutes les informations d'identification de sécurité temporaires associées au rôle, y compris les informations d'identification émises avant que vous apportiez des changements à la politique d'autorisations du rôle.

Après avoir mis à jour la politique, vous pouvez révoquer les informations d'identification de sécurité temporaires du rôle pour révoquer immédiatement toutes les autorisations relatives aux informations d'identification émises pour le rôle.

Refus d'accès à une session de rôle IAM spécifique

Lorsque vous mettez à jour les rôles IAM à l'aide d'une politique de refus ou que vous supprimez complètement le rôle, tous les utilisateurs ayant accès au rôle sont perturbés. Vous pouvez refuser l'accès sans affecter les autorisations de toutes les autres sessions associées au rôle.

Le `Principal` peut se voir refuser des autorisations à l'aide des clés de contexte de condition ou des politiques basées sur les ressources.

> **Tip**
>
> Vous pouvez trouver les utilisateurs ARNs fédérés à l'aide AWS CloudTrail des journaux. Pour plus d'informations, consultez Comment identifier facilement vos utilisateurs fédérés à l'aide AWS CloudTrail de.

Refus d'accès aux sessions utilisant des informations d'identification de sécurité temporaires avec des clés de contexte de condition

Vous pouvez utiliser des clés de contexte de condition dans les politiques basées sur l'identité dans les situations où vous souhaitez refuser l'accès à des sessions d'informations d'identification de sécurité temporaires spécifiques sans affecter les autorisations de l'utilisateur ou du rôle IAM qui a créé les informations d'identification. Pour les rôles IAM, après avoir mis à jour la politique, vous pouvez également révoquer les sessions d'informations d'identification temporaires du rôle pour révoquer immédiatement toutes les informations d'identification émises.

Pour plus d'informations sur les clés de contexte de condition, veuillez consulter la rubrique AWS clés de contexte de condition globale.

lois : PrincipalArn

Vous pouvez utiliser la clé de contexte de condition aws : PrincipalArn dans une politique basée sur l'identité pour refuser l'accès à un principal spécifique en utilisant son Amazon Resource Name (ARN). Pour ce faire, spécifiez l'ARN de l'utilisateur, du rôle ou de la session utilisateur AWS STS fédérée IAM auxquels les informations d'identification de sécurité temporaires sont associées dans l'élément Condition d'une politique.

Pour refuser l'accès à un principal spécifique par son ARN

1. Dans le volet de navigation de la console IAM, choisissez Utilisateurs ou Rôles.
2. Choisissez le nom de l'utilisateur ou du rôle IAM à modifier. Vous pouvez utiliser la zone de recherche pour filtrer la liste.
3. Sélectionnez l'onglet Autorisations.
4. Sélectionnez la politique concernée à modifier. Avant de modifier une politique gérée par le client, consultez l'onglet Entités attachées pour éviter de perturber l'accès à d'autres identités auxquelles la même politique est attachée.
5. Choisissez l'onglet JSON et ajoutez une instruction de refus pour l'ARN principal, comme indiqué dans l'exemple suivant.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:role/ROLENAME",
            "arn:aws:iam::222222222222:user/USERNAME",
            "arn:aws:iam::222222222222:federated-user/USERNAME"
          ]
        }
      }
    }
```

```
  ]
}
```

6. Sur la page Vérification, vérifiez le Récapitulatif de la politique, puis choisissez Enregistrer les modifications pour enregistrer votre travail.

lois : SourceIdentity

Vous pouvez utiliser la clé de contexte de condition aws : SourceIdentity dans une politique basée sur l'identité pour refuser l'accès à une identité source spécifique associée à une session de rôle IAM. Cela s'applique tant que la session de rôle a été émise en définissant le paramètre de `SourceIdentity` demande lorsque le principal a assumé un rôle à l'aide de AWS STS `assume-role` commandes* CLI ou d'opérations d'API AWS STS `AssumeRole` *. Pour cela, vous spécifiez l'identité source à laquelle les informations d'identification de sécurité temporaires sont associées dans l'élément `Condition` d'une politique.

Contrairement à la clé de contexte sts:RoleSessionName, une fois l'identité source définie, la valeur ne peut plus être modifiée. Elle clé `aws:SourceIdentity` est présente dans le contexte de la requête pour toutes les actions entreprises par le rôle. L'identité source persiste dans les sessions de rôle suivantes lorsque vous utilisez les informations d'identification de la session pour assumer un autre rôle. Le fait d'endosser un rôle à partir d'un autre est appelé chaînage des rôles.

La politique suivante montre un exemple de la manière dont vous pouvez refuser l'accès à des sessions d'informations d'identification de sécurité temporaires à l'aide d'une clé de contexte de condition `aws:SourceIdentity`. Si vous spécifiez l'identité source associée à une session de rôle, elle interdira les sessions de rôle avec l'identité source nommée sans affecter les autorisations du rôle qui a créé les informations d'identification. Dans cet exemple, l'identité source définie par le principal lors de l'émission de la session de rôle est `nikki_wolf@example.com`. Toute requête effectuée par une session de rôle avec l'identité source `nikki_wolf@example.com` sera refusée, car l'identité source est incluse dans la condition de politique et l'effet de politique est défini sur `Deny`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
```

```
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:SourceIdentity": [
            "nikki_wolf@example.com",
            "<source identity value>"
          ]
        }
      }
    }
  ]
}
```

aws:userid

Vous pouvez utiliser la clé de contexte de condition aws:userid dans une politique basée sur l'identité pour interdire l'accès à toutes les sessions d'informations d'identification de sécurité temporaires ou à des sessions spécifiques associées à l'utilisateur ou au rôle IAM. Pour ce faire, spécifiez l'identifiant unique (ID) de l'utilisateur, du rôle ou de la session utilisateur AWS STS fédérée IAM auxquels les informations d'identification de sécurité temporaires sont associées dans l'`Condition`élément d'une politique.

La politique suivante montre un exemple de la manière dont vous pouvez refuser l'accès à des sessions d'informations d'identification de sécurité temporaires à l'aide d'une clé de contexte de condition `aws:userid`.

- `AIDAXUSER1` représente l'ID unique d'un utilisateur IAM. La spécification de l'identifiant unique d'un utilisateur IAM en tant que valeur de la clé de contexte `aws:userid` aura pour effet de refuser l'accès à l'utilisateur IAM. Cela inclut toutes les sessions d'informations d'identification de sécurité temporaires créées lors de l'appel de l'API `GetSessionToken`.
- `AROAXROLE1:*` représente l'ID unique de toutes les sessions associées au rôle IAM. La spécification de l'ID unique d'un rôle IAM et d'un caractère générique (*) dans la partie caller-specified-role-session -name comme valeur de clé de contexte `aws:userid` empêchera toutes les sessions associées au rôle.
- `AROAXROLE2:<caller-specified-role-session-name>` représente l'ID unique d'une session de rôle assumé. Dans la partie caller-specified-role-session -name de l'ID unique du rôle assumé, vous pouvez spécifier un rôle, un nom de session ou un caractère générique si l' StringLike opérateur de condition est utilisé. Si vous spécifiez le nom de la session de rôle,

cela refusera la session de rôle nommée sans affecter les autorisations du rôle qui a créé les informations d'identification. Si vous spécifiez un caractère générique pour le nom de session du rôle, toutes les sessions associées au rôle seront refusées.

> **Note**
>
> Le nom de session de rôle caller-specified, qui fait partie de l'identifiant unique d'une session assumed-role, peut changer pendant le chaînage des rôles. Le chaînage des rôles se produit lorsqu'un rôle assume un autre rôle. Le nom de session du rôle est défini à l'aide du paramètre de `RoleSessionName` requête lorsque le principal assume un rôle à l'aide de l'opération AWS STS `AssumeRole` API.

- `account-id:<federated-user-caller-specified-name>`représente l'ID unique d'une session utilisateur AWS STS fédérée. Un utilisateur IAM crée cette session en appelant l'API `GetFederationToken`. La spécification de l'ID unique d'une session d'utilisateur fédéré AWS STS a pour effet de refuser la session fédérée nommée sans affecter les autorisations de l'utilisateur IAM qui a créé les informations d'identification.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:userId": [
            "AIDAXUSER1",
            "AROAXROLE1:*",
            "AROAXROLE2:<caller-specified-role-session-name>",
            "123456789012:<federated-user-caller-specified-name>"
          ]
        }
      }
    }
  ]
```

```
}
```

Pour des exemples spécifiques de valeurs de clé de principal, veuillez consulter la rubrique Valeurs de la clé du principal. Pour plus d'informations sur les identifiants uniques IAM et sur la manière de les obtenir, consultez Identifiants uniques.

Refus d'accès à un principal spécifique avec des politiques basées sur les ressources

Pour restreindre l'accès à un principal spécifique à l'aide d'une politique basée sur les ressources, vous pouvez utiliser des clés contextuelles de condition aws : PrincipalArn ou aws : SourceIdentity dans l'élément `Condition`. Une politique basée sur les ressources est une politique d'autorisation attachée à une ressource et qui contrôle qui peut accéder à la ressource et quelles actions il peut effectuer sur celle-ci.

Lorsque vous utilisez la clé de `aws:PrincipalARN` contexte, spécifiez l'ARN de l'utilisateur, du rôle ou de la session utilisateur AWS STS fédérée IAM associé aux informations d'identification de sécurité temporaires dans l'élément Condition d'une politique. L'exemple de politique suivant montre comment utiliser la clé de contexte `aws:PrincipalARN` dans une politique basée sur les ressources :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Principal": "*",
    "Effect": "Deny",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
    "Condition": {
      "ArnEquals": {
        "aws:PrincipalArn": [
          "arn:aws:iam::222222222222:role/ROLENAME",
          "arn:aws:iam::222222222222:user/USERNAME",
          "arn:aws:sts::222222222222:federated-user/USERNAME"
        ]
      }
    }
  }
}
```

Lorsque vous utilisez la clé de contexte `aws:SourceIdentity`, spécifiez la valeur d'identité source associée aux informations de sécurité temporaires du rôle dans l'élément `Condition` d'une politique. Cela s'applique tant que la session de rôle a été émise en définissant le paramètre de `SourceIdentity` demande lorsque le principal a assumé un rôle à l'aide de AWS STS `assume-role` commandes* CLI ou d'opérations d'API AWS STS `AssumeRole` *. L'exemple suivant montre comment utiliser la clé de contexte `aws:SourceIdentity` dans une politique basée sur une ressource :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Principal": "*",
    "Effect": "Deny",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
    "Condition": {
      "StringLike": {
        "aws:SourceIdentity": [
          "nikki_wolf@example.com",
          "<source identity value>"
        ]
      }
    }
  }
}
```

Si vous mettez uniquement à jour la politique basée sur l'identité d'un principal, celui-ci peut toujours effectuer les actions autorisées dans la politique basée sur les ressources, sauf lorsque ces actions sont explicitement refusées dans la politique basée sur l'identité.

Pour refuser l'accès à un principal spécifique dans une politique basée sur les ressources

1. Reportez-vous à la rubrique AWS services qui fonctionnent avec IAM pour voir si le service prend en charge les politiques basées sur les ressources.
2. Connectez-vous à la console du service Console de gestion AWS et ouvrez-la. Chaque service dispose d'un emplacement différent dans la console pour associer des politiques.

3. Modifiez la politique basée sur une ressource. Ajoutez une instruction de politique de refus pour spécifier les informations d’identification de l’identifiant :

   a. Dans l’élément `Principal`, saisissez le caractère générique (*). Le principal sera restreint dans l’élément `Condition`.

   b. Dans l’élément `Effect`, saisissez « Deny ».

   c. Dans `Action`, saisissez l'espace de noms du service et le nom de l'action à refuser. Pour refuser toutes les actions, utilisez le caractère générique (*). Par exemple : `"s3:*"`.

   d. Dans l’élément `Resource`, saisissez l’ARN de la ressource cible. Par exemple : `"arn:aws:s3:::amzn-s3-demo-bucket"`.

   e. Dans l’élément `Condition`, spécifiez la clé de contexte `aws:PrincipalARN` ou `aws:SourceIdentity`.

      Si vous utilisez la clé de contexte `aws:PrincipalARN`, saisissez l’ARN du principal pour lequel refuser l’accès.

      Si vous utilisez la clé de contexte `aws:SourceIdentity`, saisissez la valeur d’identité source définie dans la session de rôle pour laquelle refuser l’accès.

4. Enregistrez votre travail.

Octroi d'autorisations pour créer des informations d'identification de sécurité temporaires

Par défaut, les utilisateurs IAM n’ont pas l’autorisation de créer des informations d’identification de sécurité temporaires pour des sessions d’utilisateurs fédérés et les rôles AWS STS . Vous devez utiliser une politique pour fournir ces autorisations à vos utilisateurs. Bien que vous puissiez accorder des autorisations directement à un utilisateur, nous vous recommandons vivement d'accorder des autorisations à un groupe. Cela facilite la gestion des autorisations. Lorsqu'un utilisateur n'a plus besoin d'effectuer les tâches associées aux autorisations, il vous suffit de le supprimer du groupe. Si un autre utilisateur doit effectuer cette tâche, ajoutez-le au groupe pour lui accorder les autorisations.

Pour accorder à un groupe IAM l’autorisation de créer des informations d’identification de sécurité temporaires pour des sessions d’utilisateurs fédérés ou des rôles AWS STS , vous attachez une politique qui accorde un ou plusieurs des privilèges suivants :

- Pour que les principaux fédérés OIDC et SAML puissent accéder à un rôle IAM, accordez l'accès à. AWS STS `AssumeRole`

- Pour les utilisateurs AWS STS fédérés qui n'ont pas besoin de rôle, accordez l'accès à AWS STS `GetFederationToken`.

Pour plus d'informations sur les différences entre les opérations d'API `AssumeRole` et `GetFederationToken`, consultez Demande d'identifiants de sécurité temporaires.

Les utilisateurs IAM peuvent également appeler `GetSessionToken` pour créer des informations d'identification de sécurité temporaires. Aucune autorisation n'est requise pour qu'un utilisateur puisse appeler `GetSessionToken`. Le but principal de cette opération est d'authentifier l'utilisateur à l'aide de l'authentification MFA. Vous ne pouvez pas utiliser les politiques pour contrôler l'authentification. Cela signifie que vous ne pouvez pas empêcher les utilisateurs IAM d'appeler `GetSessionToken` pour créer des informations d'identification temporaires.

Example Exemple de politique qui accorde l'autorisation d'endosser un rôle

L'exemple de politique suivant accorde l'autorisation d'appeler `AssumeRole` pour le `UpdateApp` rôle dans Compte AWS `123123123123`. Quand `AssumeRole` est utilisé, l'utilisateur (ou l'application) qui crée les informations d'identification de sécurité pour le compte d'un utilisateur fédéré ne peut déléguer aucune autorisation non spécifiée dans la politique d'autorisation de rôle.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::123123123123:role/UpdateAPP"
  }]
}
```

Example Exemple de politique qui accorde l'autorisation de créer des informations d'identification de sécurité temporaires pour un utilisateur fédéré

L'exemple de politique suivant donne l'autorisations d'accéder à `GetFederationToken`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": "*"
  }]
}
```

> **Important**
>
> Lorsque vous accordez à un utilisateur IAM l'autorisation de créer des informations d'identification de sécurité temporaires pour des utilisateurs fédérés AWS STS avec `GetFederationToken`, sachez que cela lui autorise à déléguer ses propres autorisations. Pour plus d'informations sur la délégation d'autorisations entre les utilisateurs IAM et consultez Comptes AWS. Exemples de politiques pour la délégation d'accès Pour plus d'informations sur le contrôle des autorisations dans les informations d'identification de sécurité temporaires, consultez Autorisations affectées aux informations d'identification de sécurité temporaires.

Example Exemple de politique qui accorde à un utilisateur une autorisation limitée de créer des informations d'identification de sécurité temporaires pour des utilisateurs fédérés

Lorsque vous laissez un utilisateur IAM appeler `GetFederationToken`, il s'agit d'une bonne pratique pour limiter les autorisations que l'utilisateur IAM peut déléguer. Par exemple, la politique suivante indique comment autoriser un utilisateur IAM à créer des informations d'identification de sécurité temporaires uniquement pour les utilisateurs AWS STS fédérés dont le nom commence par Manager.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
```

```
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": ["arn:aws:sts::123456789012:federated-user/Manager*"]
  }]
}
```

Octroi d'autorisations pour l'utilisation de sessions de console à identité renforcée

Les sessions de console à identité améliorée permettent AWS IAM Identity Center d'inclure l'utilisateur et IDs la session dans les sessions de AWS console des utilisateurs lorsqu'ils se connectent. Par exemple, Amazon Q Developer Pro utilise des sessions de console à identité renforcée pour personnaliser l'expérience de service. Pour plus d'informations sur les sessions de console à identité renforcée, consultez la section Activation des sessions de console à identité renforcée dans le Guide de l'utilisateur AWS IAM Identity Center . Pour plus d'informations sur la configuration d'Amazon Q Developer, consultez la section Configuration d'Amazon Q Developer dans le Guide de l'utilisateur Amazon Q Developer.

Pour qu'un utilisateur puisse disposer de sessions de console à identité renforcée, vous devez utiliser une politique basée sur l'identité pour accorder au principal IAM l'autorisation `sts:SetContext` pour la ressource qui représente sa propre session de console.

> **Important**
>
> Par défaut, les utilisateurs n'ont pas l'autorisation de définir le contexte de leurs sessions de console à identité renforcée. Pour ce faire, vous devez accorder l'autorisation `sts:SetContext` au principal IAM dans une politique basée sur l'identité, comme indiqué dans l'exemple de politique ci-dessous.

L'exemple de politique basée sur l'identité suivant accorde l'`sts:SetContext`autorisation à un principal IAM, ce qui permet au principal de définir un contexte de session de console à identité améliorée pour ses propres sessions de console. AWS La ressource de politique représente la AWS session de l'appelant. `arn:aws:sts::`*`account-id`*`:self` Le segment ARN `account-id` peut être remplacé par un caractère générique `*` dans les cas où la même politique d'autorisation est déployée sur plusieurs comptes, par exemple lorsque cette politique est déployée à l'aide des ensembles d'autorisations d'IAM Identity Center.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:SetContext",
            "Resource": "arn:aws:sts::111122223333:self"
        }
    ]
}
```

Gérer AWS STS dans un Région AWS

Un point de terminaison régional est l'URL du point d'entrée d'un service AWS Web dans une région donnée. AWS recommande d'utiliser les points de terminaison régionaux AWS Security Token Service (AWS STS) au lieu du point de terminaison global pour réduire la latence, intégrer la redondance et augmenter la validité des jetons de session.

Bien que le point de AWS STS terminaison mondial (ancien) soit hautement disponible, il `https://sts.amazonaws.com` est hébergé dans une seule AWS région, l'est des États-Unis (Virginie du Nord), et comme les autres points de terminaison, il ne propose pas de basculement automatique vers les points de terminaison des autres régions.

- Réduire la latence : en passant vos AWS STS appels vers un point de terminaison géographiquement plus proche de vos services et applications, vous pouvez accéder à des AWS STS services avec une latence plus faible et de meilleurs temps de réponse.
- Intégrer de la redondance : vous pouvez limiter les effets d'une défaillance au sein d'une charge de travail à un nombre limité de composants avec une portée prévisible de maîtrise des impacts. L'utilisation de AWS STS points de terminaison régionaux vous permet d'aligner la portée de vos composants sur celle de vos jetons de session. Pour plus d'informations sur ce pilier de fiabilité, voir Utiliser l'isolation des pannes pour protéger votre charge de travail dans le AWS Well-Architected Framework.
- Augmentez la validité des jetons de session — Les jetons de session des AWS STS terminaux régionaux sont valides pour tous Régions AWS. Les jetons de session du point de terminaison STS global ne sont valides Régions AWS que s'ils sont activés par défaut. Si vous avez l'intention

d'activer une nouvelle région pour votre compte, vous pouvez utiliser des jetons de session depuis les AWS STS points de terminaison régionaux. Si vous choisissez d'utiliser le point de terminaison global, vous devez modifier la compatibilité régionale des jetons de AWS STS session pour le point de terminaison global. Cela garantit la validité des jetons dans tous les cas Régions AWS.

Pour une liste des AWS STS régions et de leurs points de terminaison, consultezAWS STS Régions et points de terminaison.

> **Note**
>
> AWS a apporté des modifications au AWS Security Token Service point de terminaison global (`https://sts.amazonaws.com`) dans les régions activé par défaut afin d'améliorer sa résilience et ses performances.AWS STS AWS STS les demandes adressées au point de terminaison global sont automatiquement traitées de la même manière Région AWS que vos charges de travail. Ces modifications ne seront pas déployées vers des régions d'adhésion. Nous vous recommandons d'utiliser les points de terminaison AWS STS régionaux appropriés. Pour de plus amples informations, veuillez consulter AWS STS modifications du point de terminaison global.

Rubriques

- Activation et désactivation AWS STS dans un Région AWS
- Écrire du code à utiliser AWS STS Régions
- Gestion des jetons de session de point de terminaison global

Activation et désactivation AWS STS dans un Région AWS

Lorsque vous activez des AWS STS points de terminaison pour une région, vous AWS STS pouvez délivrer des informations d'identification temporaires aux utilisateurs et aux rôles de votre compte qui en font la AWS STS demande. Ces informations d'identification peuvent ensuite être utilisées dans n'importe quelle région activée par défaut ou activée manuellement.

Pour les régions activées par défaut, vous devez activer le point de AWS STS terminaison régional dans le compte sur lequel les informations d'identification temporaires sont générées. Peu importe si un utilisateur est connecté au même compte ou à un compte différent lorsqu'il fait la demande.

Lorsque vous demandez des informations d'identification temporaires pour un rôle dans une autre région à Compte AWS l'aide d'une région activée manuellement, le compte cible (le compte contenant le rôle) doit activer cette région pour les AWS STS opérations. Cela garantit que les informations d'identification de sécurité temporaires peuvent être générées correctement.

Par exemple, un utilisateur du compte A veut envoyer une demande d'API `sts:AssumeRole` au point de terminaison AWS STS régional `https://sts.ap-southeast-3.amazonaws.com`. La demande concerne les informations d'identification temporaires du rôle nommé `Developer` dans le compte B. Étant donné qu'il s'agit d'une demande de création d'informations d'identification pour une entité du compte B, la région `ap-southeast-3` doit être activée pour le compte B. Les utilisateurs du compte A (ou de tout autre compte) peuvent appeler le point de terminaison `ap-southeast-3` AWS STS pour demander les informations d'identification pour le compte B, que la région soit activée ou non dans leurs comptes. Pour en savoir plus, consultez la section Activer ou désactiver Régions AWS dans votre compte.

> **Note**
>
> Les régions actives sont disponibles pour chaque personne qui utilise des informations d'identification temporaires de ce compte. Pour contrôler les utilisateurs ou rôles IAM qui peuvent accéder à la région, utilisez la clé de condition `aws:RequestedRegion` dans vos politiques d'autorisations.

Pour activer ou désactiver AWS STS dans une région activée par défaut (console)

1. Connectez-vous en tant qu'utilisateur root ou utilisateur IAM avec les autorisations pour effectuer des tâches d'administration d'IAM.
2. Ouvrez la console IAM, puis dans le panneau de navigation, sélectionnez Account settings (Paramètres du compte).
3. Dans la section Security Token Service (STS) Points de terminaison, recherchez la région que vous souhaitez configurer, puis choisissez Active ou Inactive dans la colonne d'état STS.
4. Dans la boîte de dialogue qui s'ouvre, choisissez Activate ou Deactivate (Activer ou Désactiver).

Pour les régions qui doivent être activées, nous les activons AWS STS automatiquement lorsque vous activez la région. Une fois que vous avez activé une région, AWS STS il est toujours actif pour cette région et vous ne pouvez pas le désactiver. Pour en savoir plus sur l'activation des régions

désactivées par défaut, consultez la section Spécifier celles que Régions AWS votre compte peut utiliser dans le Guide de Gestion de compte AWS référence.

Écrire du code à utiliser AWS STS Régions

Après avoir activé une région, vous pouvez diriger les appels d' AWS STS API vers cette région. L'extrait de code Java suivant illustre comment configurer un objet `AWSSecurityTokenService` pour effectuer des requêtes vers la région Europe (Milan) (eu-south-1).

```
EndpointConfiguration regionEndpointConfig = new EndpointConfiguration("https://sts.eu-
south-1.amazonaws.com", "eu-south-1");
AWSSecurityTokenService stsRegionalClient =
 AWSSecurityTokenServiceClientBuilder.standard()
.withCredentials(credentials)
.withEndpointConfiguration(regionEndpointConfig)
.build();
```

AWS STS vous recommande de passer des appels vers un point de terminaison régional. Pour savoir comment activer manuellement une région, consultez la section Spécification des Régions AWS que votre compte peut utiliser dans le Guide de référence Gestion de compte AWS .

Dans l'exemple, la première ligne instancie un objet `EndpointConfiguration` appelé `regionEndpointConfig`, en transmettant l'URL du point de terminaison et la Région AWS comme paramètres.

Pour savoir comment définir des points de terminaison AWS STS régionaux à l'aide d'une variable d'environnement pour les AWS SDK, consultez la section Points de terminaison AWS STS régionalisés dans le Guide de référence des AWS kits de développement logiciel et des outils.

Pour toutes les autres combinaisons de langage et d'environnement de programmation, reportez-vous à la documentation du kit SDK approprié.

Gestion des jetons de session de point de terminaison global

La plupart des opérations Régions AWS sont activées Services AWS par défaut. Ces régions sont automatiquement activées pour être utilisées avec AWS STS. Certaines régions, telles que l'Asie-Pacifique (Hong Kong), doivent être activées manuellement. Pour en savoir plus sur l'activation et la désactivation des Régions AWS, consultez la section Spécification des Régions AWS que votre compte peut utiliser dans le Guide de référence Gestion de compte AWS .

Lorsque vous activez ces AWS régions, elles sont automatiquement activées pour être utilisées avec AWS STS. Vous ne pouvez pas activer le AWS STS point de terminaison pour une région désactivée. Les jetons de session valides dans tous les cas Régions AWS comportent plus de caractères que les jetons valides dans les régions activées par défaut. La modification de ce paramètre peut avoir un impact sur les systèmes existants où vous stockez temporairement les jetons.

Vous pouvez modifier ce paramètre à l'aide de l'API Console de gestion AWS AWS CLI, ou de AWS l'API.

Pour modifier la compatibilité régionale des jetons de session pour le point de terminaison global (console)

1. Connectez-vous en tant qu'utilisateur root ou utilisateur IAM avec les autorisations pour effectuer des tâches d'administration d'IAM. Pour modifier la compatibilité des jetons de session, vous devez posséder une politique qui autorise l'action `iam:SetSecurityTokenServicePreferences`.
2. Ouvrez la console IAM. Dans le panneau de navigation, choisissez Paramètres du compte.
3. Dans la section Security Token Service (STS), Jetons de session provenant des points de terminaison STS. Le point de terminaison global indique `Valid only in Régions AWS enabled by default`. Choisissez Change (Modifier).
4. Dans la boîte de dialogue Modifier la compatibilité des régions, sélectionnez Tout Régions AWS. Ensuite, choisissez Enregistrer les modifications.

> **Note**
>
> Les jetons de session valides dans tous les cas Région AWS comportent plus de caractères que les jetons valides dans les régions activées par défaut. La modification de ce paramètre peut avoir un impact sur les systèmes existants où vous stockez temporairement les jetons.

Pour modifier la compatibilité régionale des jetons de session pour le point de terminaison global (AWS CLI)

Définissez la version du jeton de session. Les jetons de version 1 ne sont valides Régions AWS que s'ils sont disponibles par défaut. Ces jetons ne fonctionnent pas dans les régions activées manuellement, par exemple, Asie-Pacifique (Hong Kong). Les jetons de la version 2 sont valides

dans toutes les régions. Toutefois, les jetons de version 2 incluent plus de caractères et peuvent avoir un impact sur les systèmes où vous stockez temporairement les jetons.

- `aws iam set-security-token-service-preferences`

Pour modifier la compatibilité régionale des jetons de session pour le point de terminaison global (AWS API)

Définissez la version du jeton de session. Les jetons de version 1 ne sont valides Régions AWS que s'ils sont disponibles par défaut. Ces jetons ne fonctionnent pas dans les régions activées manuellement, par exemple, Asie-Pacifique (Hong Kong). Les jetons de la version 2 sont valides dans toutes les régions. Toutefois, les jetons de version 2 incluent plus de caractères et peuvent avoir un impact sur les systèmes où vous stockez temporairement les jetons.

- `SetSecurityTokenServicePreferences`

AWS STS Régions et points de terminaison

> **Note**
>
> AWS a apporté des modifications au AWS Security Token Service point de terminaison global (`https://sts.amazonaws.com`) dans les régions activées par défaut afin d'améliorer sa résilience et ses performances.AWS STS AWS STS les demandes adressées au point de terminaison global sont automatiquement traitées au même Région AWS titre que vos charges de travail. Ces modifications ne seront pas déployées vers des régions d'adhésion. Nous vous recommandons d'utiliser les points de terminaison AWS STS régionaux appropriés. Pour de plus amples informations, veuillez consulter AWS STS modifications du point de terminaison global.

Le tableau suivant répertorie les régions et leurs points de terminaison. Il indique les régions activées par défaut et celles que vous pouvez activer ou désactiver.

Nom de la région	Endpoint	Activé par défaut	Activer/désactiver manuellement
--Global--	sts.amazonaws.com	✓ Oui	✗ Non
USA Est (Ohio)	sts.us-east-2.amazonaws.com	✓ Oui	✓ Oui
USA Est (Virginie du Nord)	sts.us-east-1.amazonaws.com	✓ Oui	✗ Non
USA Ouest (Californie du Nord)	sts.us-west-1.amazonaws.com	✓ Oui	✓ Oui
USA Ouest (Oregon)	sts.us-west-2.amazonaws.com	✓ Oui	✓ Oui
Afrique (Le Cap)	sts.af-south-1.amazonaws.com	✗ Non[1]	✗ Non
Asie-Pacifique (Hong Kong)	sts.ap-east-1.amazonaws.com	✗ Non[1]	✗ Non
Asie-Pacifique (Hyderabad)	sts.ap-south-2.amazonaws.com	✗ Non[1]	✗ Non
Asie-Pacifique (Jakarta)	sts.ap-southeast-3.amazonaws.com	✗ Non[1]	✗ Non
Asie-Pacifique (Malaisie)	sts.ap-southeast-5.amazonaws.com	✗ Non[1]	✗ Non
Asie-Pacifique (Melbourne)	sts.ap-southeast-4.amazonaws.com	✗ Non[1]	✗ Non
Asie-Pacifique (Mumbai)	sts.ap-south-1.amazonaws.com	✓ Oui	✓ Oui

Nom de la région	Endpoint	Activé par défaut	Activer/désactiver manuellement
Asie-Pacifique (Osaka)	sts.ap-northeast-3.amazonaws.com	✓ Oui	✓ Oui
Asie-Pacifique (Séoul)	sts.ap-northeast-2.amazonaws.com	✓ Oui	✓ Oui
Asie-Pacifique (Singapour)	sts.ap-southeast-1.amazonaws.com	✓ Oui	✓ Oui
Asie-Pacifique (Sydney)	sts.ap-southeast-2.amazonaws.com	✓ Oui	✓ Oui
Asie-Pacifique (Thaïlande)	sts.ap-southeast-7.amazonaws.com	✗ Non[1]	✗ Non
Asie-Pacifique (Tokyo)	sts.ap-northeast-1.amazonaws.com	✓ Oui	✓ Oui
Canada (Centre)	sts.ca-central-1.amazonaws.com	✓ Oui	✓ Oui
Canada-Ouest (Calgary)	sts.ca-west-1.amazonaws.com	✗ Non[1]	✗ Non
Chine (Pékin)	sts.cn-north-1.amazonaws.com.cn	✓ Oui[1]	✗ Non
Chine (Ningxia)	sts.cn-northwest-1.amazonaws.com.cn	✓ Oui[1]	✓ Oui
Europe (Francfort)	sts.eu-central-1.amazonaws.com	✓ Oui	✓ Oui
Europe (Irlande)	sts.eu-west-1.amazonaws.com	✓ Oui	✓ Oui
Europe (Londres)	sts.eu-west-2.amazonaws.com	✓ Oui	✓ Oui
Europe (Milan)	sts.eu-south-1.amazonaws.com	✗ Non[1]	✗ Non

Nom de la région	Endpoint	Activé par défaut	Activer/désactiver manuellement
Europe (Paris)	sts.eu-west-3.amazonaws.com	✓ Oui	✓ Oui
Europe (Espagne)	sts.eu-south-2.amazonaws.com	✗ Non[1]	✗ Non
Europe (Stockholm)	sts.eu-north-1.amazonaws.com	✓ Oui	✓ Oui
Europe (Zurich)	sts.eu-central-2.amazonaws.com	✗ Non[1]	✗ Non
Israël (Tel Aviv)	sts.il-central-1.amazonaws.com	✗ Non[1]	✗ Non
Mexique (Centre)	sts.mx-central-1.amazonaws.com	✗ Non[1]	✗ Non
Middle East (Bahrain)	sts.me-south-1.amazonaws.com	✗ Non[1]	✗ Non
Moyen-Orient (EAU)	sts.me-central-1.amazonaws.com	✗ Non[1]	✗ Non
Amérique du Sud (São Paulo)	sts.sa-east-1.amazonaws.com	✓ Oui	✓ Oui

[1]Vous devez activer la région pour l'utiliser. Cela active automatiquement AWS STS. Vous ne pouvez pas activer ou désactiver manuellement AWS STS dans ces régions.

[2]Pour l'utiliser AWS en Chine, vous avez besoin d'un compte et d'informations d'identification spécifiques AWS à la Chine.

AWS STS modifications du point de terminaison global

AWS a apporté des modifications au AWS Security Token Service point de terminaison global (`https://sts.amazonaws.com`) dans les régions activées par défaut afin d'améliorer sa résilience et ses performances.AWS STS Auparavant, toutes les demandes adressées au point de terminaison AWS STS mondial étaient traitées par un seul terminal Région AWS, l'est des États-Unis (Virginie du Nord). Désormais, dans les régions activées par défaut, les demandes adressées au point de terminaison AWS STS mondial sont automatiquement traitées dans la même région d'où provient la

demande, plutôt que dans la région de l'est des États-Unis (Virginie du Nord). Ces modifications ne seront pas déployées vers des régions d'adhésion.

Avec cette modification, votre demande AWS STS sera traitée en fonction de la région d'origine et du résolveur DNS utilisés. Les demandes adressées au point de terminaison AWS STS mondial sont traitées dans la même région que votre charge de travail AWS déployée si la demande DNS pour le point de terminaison AWS STS mondial est traitée par le serveur Amazon DNS dans les régions activées par défaut. Les demandes adressées au point de terminaison global AWS STS continueront d'être traitées dans la région USA Est (Virginie du Nord) si votre demande provient de régions d'adhésion ou si votre demande a été résolue à l'aide d'un résolveur DNS autre que le serveur DNS Amazon. Pour plus d'informations sur Amazon DNS, veuillez consulter Serveur Amazon DNS dans le Guide de l'utilisateur Amazon Virtual Private Cloud.

Le tableau suivant montre comment les demandes adressées au point de terminaison AWS STS global sont acheminées en fonction de votre fournisseur DNS.

Résolveur DNS	Les demandes adressées au point de terminaison AWS STS global sont-elles acheminées vers le point de terminaison local ? Région AWS
Résolveur DNS Amazon dans un VPC Amazon dans une région activée par défaut	Oui
Résolveur DNS Amazon dans un VPC Amazon dans une région d'adhésion	Non, la demande sera transmise à la région USA Est (Virginie du Nord)
Résolveur DNS fourni par votre FAI, un fournisseur DNS public ou tout autre fournisseur DNS	Non, la demande sera transmise à la région USA Est (Virginie du Nord)

Pour garantir une perturbation minimale de vos processus existants, AWS a mis en œuvre les mesures suivantes :

- AWS CloudTrail les journaux des demandes adressées au point de terminaison AWS STS mondial sont envoyés à la région USA Est (Virginie du Nord). CloudTrail les journaux des demandes traitées par les points de terminaison AWS STS régionaux continueront d'être enregistrés dans CloudTrail leur région respective.

- CloudTrail les journaux des opérations effectuées par le point de terminaison AWS STS global et les points de terminaison régionaux comportent des champs supplémentaires `endpointType` et `awsServingRegion` indiquent quel point de terminaison et quelle région ont répondu à la demande. Pour des exemples de CloudTrail journaux, voirExemple AWS STS Événement d'API utilisant le point de terminaison global dans le fichier CloudTrail journal.
- Les demandes adressées au point de terminaison AWS STS global ont une valeur égale à `us-east-1` pour la clé de `aws:RequestedRegion` condition, quelle que soit la région ayant répondu à la demande.
- Les demandes traitées par le point de terminaison AWS STS global ne partagent pas de quota de demandes par seconde avec les AWS STS points de terminaison régionaux.

Si vous avez des charges de travail dans une région optionnelle et que vous utilisez toujours le point de terminaison AWS STS mondial, nous vous recommandons de migrer vers des points de terminaison AWS STS régionaux pour améliorer la résilience et les performances. Pour plus d'informations sur la configuration des points de AWS STS terminaison régionaux, consultez la section Points de terminaison AWS STS régionaux dans le guide de référence des outils AWS SDKs et.

AWS CloudTrail et points de terminaison régionaux

Les appels vers les points de terminaison régionaux et globaux sont enregistrés dans le champ `tlsDetails` dans AWS CloudTrail. Les appels vers les points de terminaison régionaux`us-east-2.amazonaws.com`, tels que, sont connectés CloudTrail à la région appropriée. Les appels vers le point de terminaison global, `sts.amazonaws.com`, sont consignés en tant qu'appels à un service international. Les événements relatifs aux AWS STS points de terminaison globaux sont enregistrés dans us-east-1.

> **Note**
>
> `tlsDetails` peut uniquement être consulté pour les services qui prennent en charge ce champ. Voir les informations sur les services compatibles avec le protocole TLS CloudTrail dans le guide de l'AWS CloudTrail utilisateur
> Pour de plus amples informations, veuillez consulter Journalisation IAM et AWS STS Appels d'API avec AWS CloudTrail.

Activez l'accès personnalisé du courtier d'identité au AWS console

Vous pouvez écrire et exécuter du code afin de créer une URL qui permet aux utilisateurs qui se connectent au réseau de votre organisation d'accéder en toute sécurité à Console de gestion AWS. L'URL inclut un jeton de connexion que vous obtenez AWS et qui authentifie l'utilisateur. AWS La session de console obtenue peut inclure un `AccessKeyId` distinct en raison de la fédération. Pour suivre l'utilisation des clés d'accès pour la connexion à la fédération par le biais d' CloudTrail événements connexes, consultez la section Événements Journalisation IAM et AWS STS Appels d'API avec AWS CloudTrail de Console de gestion AWS connexion.

> **Note**
>
> Si votre organisation utilise un fournisseur d'identité (IdP) qui est compatible avec SAML, vous pouvez configurer l'accès à la console sans écrire de code. Cela fonctionne avec des fournisseurs tels que Microsoft Active Directory Federation Services ou Shibboleth (open source). Pour en savoir plus, consultez Permettre aux principaux fédérés SAML 2.0 d'accéder au Console de gestion AWS.

Pour permettre aux utilisateurs de votre organisation d'accéder au Console de gestion AWS, vous pouvez créer un courtier d'identité personnalisé qui exécute les étapes suivantes :

1. Vérifier que l'utilisateur est authentifié par votre système d'identité local.
2. Appelez les opérations AWS Security Token Service (AWS STS) AssumeRole (recommandé) ou GetFederationToken API pour obtenir des informations d'identification de sécurité temporaires pour l'utilisateur. Pour connaître les différentes méthodes que vous pouvez utiliser pour endosser un rôle, consultez Méthodes pour assumer un rôle. Pour savoir comment transmettre des balises de session facultatives lorsque vous obtenez vos informations d'identification de sécurité, veuillez consulter Transmettre les balises de session AWS STS.
   - Si vous utilisez l'une des opérations d'API `AssumeRole*` pour obtenir les informations d'identification de sécurité temporaires pour un rôle, vous pouvez inclure le paramètre `DurationSeconds` dans votre appel. Ce paramètre spécifie la durée de la session de votre rôle, entre 900 secondes (15 minutes) et la durée de session maximale pour le rôle. Lorsque vous utilisez `DurationSeconds` dans une opération `AssumeRole*`, vous devez l'appeler en tant qu'utilisateur IAM avec des informations d'identification à long terme. Sinon, l'appel au point de terminaison de fédération de l'étape 3 échoue. Pour savoir comment afficher ou modifier la

valeur maximale pour un rôle, consultez Mettre à jour la durée de session maximale pour un rôle.

- Si vous utilisez l'opération d'API `GetFederationToken` pour obtenir les informations d'identification, vous pouvez inclure le paramètre `DurationSeconds` dans votre appel. Ce paramètre spécifie la durée de votre session de rôle. La valeur peut être comprise entre 900 secondes (15 minutes) et 129 600 secondes (36 heures). Vous pouvez effectuer cet appel d'API uniquement à l'aide des informations d'identification de AWS sécurité à long terme d'un utilisateur IAM. Vous pouvez également passer ces appels à l'aide Utilisateur racine d'un compte AWS d'informations d'identification, mais nous ne le recommandons pas. Si vous effectuez cet appel en tant qu'utilisateur racine, la session par défaut dure une heure. Ou vous pouvez spécifier une session de 900 secondes (15 minutes) à 3 600 secondes (une heure).

3. Appelez le point de terminaison de la AWS fédération et fournissez les informations d'identification de sécurité temporaires pour demander un jeton de connexion.
4. Créer une URL incluant le jeton pour la console :
   - Si vous utilisez l'une des opérations d'API `AssumeRole*` dans votre URL, vous pouvez inclure le paramètre HTTP `SessionDuration`. Ce paramètre spécifie la durée de la session de la console, de 900 secondes (15 minutes) à 43 200 secondes (12 heures).
   - Si vous utilisez l'opération d'API `GetFederationToken` dans votre URL, vous pouvez inclure le paramètre `DurationSeconds`. Ce paramètre spécifie la durée de la session de console fédérée. La valeur peut être comprise entre 900 secondes (15 minutes) et 129 600 secondes (36 heures).

   > **Note**
   >
   > - Votre `SessionDuration` ne peut être supérieure ou égale à la durée maximale de la session définie pour le rôle que vous endossez. Par exemple, vous avez défini la durée de session maximale pour le rôle que vous souhaitez assumer à 5 heures. Votre paramètre `SessionDuration` peut être 16 524 secondes ou 4 heures et 59 secondes.
   > - N'utilisez pas le paramètre HTTP `SessionDuration` lorsque vous obtenez des informations d'identification temporaires avec `GetFederationToken`. L'opération échouera.
   > - L'utilisation des informations d'identification pour qu'un rôle endosse un rôle différent est appelée chaînes de rôles. Lorsque vous utilisez la création de chaînes de rôles, vos nouvelles informations d'identification sont limitées à une durée maximale

d'une heure. Lorsque vous utilisez des rôles pour accorder des autorisations aux applications qui s'exécutent sur des instances EC2, ces applications ne sont pas soumises à cette limitation.

- N'utilisez pas le paramètre HTTP `SessionDuration` lorsque vous obtenez des informations d'identification temporaires via le chaînage de rôles. L'opération échouera.

5. Donner l'URL à l'utilisateur ou appeler l'URL pour le compte de l'utilisateur.

L'URL fournie par le point de terminaison de fédération est valide pendant 15 minutes à compter de sa création. Cette durée est différente de celle (en secondes) de la session des informations d'identification de sécurité temporaires associées à l'URL. Ces informations d'identification sont valides pendant la durée spécifiée lors de leur création et ce, à compter du moment où elles sont créées.

> **Important**
>
> L'URL donne accès à vos AWS ressources via les autorisations Console de gestion AWS si vous avez activé les autorisations dans les informations d'identification de sécurité temporaires associées. Pour cette raison, vous devez considérer l'URL comme un secret. Nous vous recommandons de retourner l'URL via une redirection sécurisée, par exemple, en utilisant un code de statut de réponse HTTP 302 via une connexion SSL. Pour plus d'informations sur le code d'état de la réponse HTTP 302, consultez la RFC 2616, section 10.3.3.

Pour effectuer ces tâches, vous pouvez utiliser l'API Query HTTPS pour Gestion des identités et des accès AWS (IAM) et AWS Security Token Service (AWS STS). Vous pouvez également utiliser des langages de programmation tels que Java, Ruby ou C#, avec le kit SDK AWS approprié. Chacune de ces méthodes est décrite dans les rubriques suivantes.

Rubriques

- Exemple de code utilisant les opérations d'API Query IAM
- Exemple de code utilisant Python
- Exemple de code utilisant Java
- Exemple montrant comment créer l'URL (Ruby)

Exemple de code utilisant les opérations d'API Query IAM

Vous pouvez créer une URL qui accorde aux rôles et aux principaux fédérés un accès direct à Console de gestion AWS. Cette tâche utilise l'API de requête IAM et AWS STS HTTPS. Pour de plus amples informations sur les demandes de requête, veuillez consulter Envoi de demandes de requête.

> **Note**
>
> La procédure suivante contient des exemples de chaînes de texte. Pour faciliter la lecture, des sauts de ligne ont été ajoutés aux exemples les plus longs. Lorsque vous créez ces chaînes pour votre propre utilisation, il convient d'omettre les sauts de ligne.

Pour permettre aux rôles et aux responsables fédérés d'accéder à vos ressources depuis le Console de gestion AWS

1. Authentifiez l'utilisateur dans votre système d'identité et d'autorisation.
2. Obtenez des informations d'identification de sécurité temporaires pour l'utilisateur. Les informations d'identification temporaires incluent un ID de clé d'accès, une clé d'accès secrète et un jeton (token) de session. Pour plus d'informations sur la création d'informations d'identification temporaires, consultez Informations d'identification de sécurité temporaires dans IAM.

   Pour obtenir des informations d'identification temporaires, vous appelez l' AWS STS AssumeRoleAPI (recommandé) ou l'GetFederationTokenAPI. Pour plus d'informations sur les différences entre ces opérations d'API, consultez la section Présentation des options d'API pour déléguer en toute sécurité l'accès à votre AWS compte dans le blog consacré à la AWS sécurité.

   > **Important**
   >
   > Lorsque vous utilisez l'GetFederationTokenAPI pour créer des informations d'identification de sécurité temporaires, vous devez spécifier les autorisations que les informations d'identification accordent à l'utilisateur qui assume le rôle. Dès lors que les opérations d'API commencent par `AssumeRole*`, vous utilisez un rôle IAM pour accorder les autorisations. Pour les autres opérations d'API, le mécanisme varie en fonction de l'API. Pour en savoir plus, consultez Autorisations affectées aux informations d'identification de sécurité temporaires. De plus, si vous utilisez les opérations d'API `AssumeRole*`, vous devez les appeler en tant qu'utilisateur IAM avec des informations

d'identification à long terme. Sinon, l'appel au point de terminaison de fédération de l'étape 3 échoue.

3. Une fois que vous avez obtenu les informations d'identification de sécurité temporaires, insérez-les dans une chaîne de session JSON pour les échanger contre un jeton de connexion. L'exemple suivant montre comment encoder les informations d'identification. Vous remplacez le texte de l'espace réservé par les valeurs appropriées dans les informations d'identification reçues à l'étape précédente.

```
{"sessionId":"*** temporary access key ID ***",
"sessionKey":"*** temporary secret access key ***",
"sessionToken":"*** session token ***"}
```

4. Encodez par URL la chaîne de session de l'étape précédente. Dans la mesure où les informations que vous encodez sont sensibles, nous vous recommandons de ne pas utiliser de service web pour cette opération. Utilisez, à la place, une fonction installée en local ou une fonctionnalité de votre boîte à outils de développement pour encoder cette information de manière sécurisée. Vous pouvez utiliser la fonction `urllib.quote_plus` dans Python, la fonction `URLEncoder.encode` dans Java ou la fonction `CGI.escape` dans Ruby. Consultez les exemples dans la suite de cette rubrique.

5.

> **Note**
>
> AWS prend en charge les requêtes POST ici.

Envoyez votre demande au point de terminaison AWS de la fédération :

`https://region-code.signin.aws.amazon.com/federation`

Pour une liste des *region-code* valeurs possibles, consultez la colonne Région dans les AWS Sign-In points de . Vous pouvez éventuellement utiliser le point de terminaison AWS Sign-In de fédération par défaut :

`https://signin.aws.amazon.com/federation`

La demande doit inclure les paramètres `Action` et `Session`, et (éventuellement), si vous avez utilisé une opération d'API `AssumeRole*`, un paramètre HTTP `SessionDuration` comme illustré dans l'exemple suivant.

```
Action = getSigninToken
SessionDuration = time in seconds
Session = *** the URL encoded JSON string created in steps 3 & 4 ***
```

> **Note**
>
> Les instructions suivantes de cette étape ne fonctionnent qu'avec les requêtes GET.

Ce paramètre HTTP `SessionDuration` spécifie la durée de la session de console. Cette durée est différente de celle des informations d'identification temporaires que vous spécifiez à l'aide du paramètre `DurationSeconds`. Vous pouvez spécifier une valeur `SessionDuration` maximale de 43 200 (12 heures). Si le `SessionDuration` paramètre est absent, la session prend par défaut la durée des informations d'identification que vous avez récupérées AWS STS à l'étape 2 (qui est par défaut d'une heure). Consultez la documentation de l'API `AssumeRole` pour plus de détails sur la spécification d'une durée à l'aide du paramètre `DurationSeconds`. La possibilité de créer une session de console d'une durée supérieure à une heure fait partie intégrante de l'opération `getSigninToken` du point de terminaison de fédération.

> **Note**
>
> - Votre `SessionDuration` ne peut être supérieure ou égale à la durée maximale de la session définie pour le rôle que vous endossez. Par exemple, vous avez défini la durée de session maximale pour le rôle que vous souhaitez assumer à 5 heures. Votre paramètre `SessionDuration` peut être 16 524 secondes ou 4 heures et 59 secondes.
> - N'utilisez pas le paramètre HTTP `SessionDuration` lorsque vous obtenez des informations d'identification temporaires avec `GetFederationToken`. L'opération échouera.
> - L'utilisation des informations d'identification pour qu'un rôle endosse un rôle différent est appelée chaînes de rôles. Lorsque vous utilisez la création de chaînes de rôles, vos nouvelles informations d'identification sont limitées à une durée maximale d'une heure. Lorsque vous utilisez des rôles pour accorder des autorisations aux applications qui s'exécutent sur des instances EC2, ces applications ne sont pas soumises à cette limitation.

- N'utilisez pas le paramètre HTTP `SessionDuration` lorsque vous obtenez des informations d'identification temporaires via le chaînage de rôles. L'opération échouera.

Lorsque vous activez des sessions de console avec une durée prolongée, vous augmentez le risque d'exposition aux informations d'identification. Pour vous aider à réduire ce risque, vous pouvez immédiatement désactiver les sessions de console actives pour un rôle en choisissant Révoquer les sessions sur la page Résumé du rôle dans la console IAM. Pour plus d'informations, veuillez consulter Révocation des informations d'identification de sécurité temporaires d'un rôle IAM.

Votre demande devrait ressembler à l'exemple suivant : Des retours à la ligne ont été ajoutés ici pour faciliter la lecture, mais vous devez envoyer une chaîne composée d'une seule ligne.

```
https://signin.aws.amazon.com/federation
?Action=getSigninToken
&SessionDuration=1800
&Session=%7B%22sessionId%22%3A+%22ASIAJUMHIZPTOKTBMK5A%22%2C+%22sessionKey%22
%3A+%22LSD7LWI%2FL%2FN%2BgYpan5QFz0XUpc8s7HYjRsgcsrsm%22%2C+%22sessionToken%2
2%3A+%22FQoDYXdzEBQaDLbj3VWv2u50NN%2F3yyLSASwYtWhPnGPMNmzZFfZsL0Qd3vtYHw5A5dW
AjOsrkdPkghomIe3mJip5%2F0djDBbo7SmO%2FENDEiCdpsQKodTpleKA8xQq0CwFg6a69xdEBQT8
FipATnLbKoyS4b%2FebhnsTUjZZQWp0wXXqFF7gSm%2FMe2tXe0jzsdP0O12obez9lijPSdF1k2b5
PfGhiuyAR9aD5%2BubM0pY86fKex1qsytjvyTbZ9nXe6DvxVDcnCOhOGETJ7XFkSFdH0v%2FYR25C
UAhJ3nXIkIbG7Ucv9cOEpCf%2Fg23ijRgILIBQ%3D%3D%22%7D
```

La réponse du point de terminaison de fédération est un document JSON avec une valeur `SigninToken`. Elle sera similaire à l'exemple suivant.

```
{"SigninToken":"*** the SigninToken string ***"}
```

6.

> **Note**
>
> AWS prend en charge les requêtes POST ici.

Enfin, créez l'URL que vos utilisateurs utiliseront pour accéder à Console de gestion AWS. L'URL est la même URL de point de terminaison de fédération que celle que vous avez utilisée dans Step 5, à laquelle sont ajoutés les paramètres suivants :

```
?Action = login
&Issuer = *** the form-urlencoded URL for your internal sign-in page ***
&Destination = *** the form-urlencoded URL to the desired AWS console page ***
&SigninToken = *** the value of SigninToken received in the previous step ***
```

> **Note**
>
> Les instructions suivantes de cette étape ne fonctionnent qu'avec l'API GET.

L'exemple suivant montre comment se présentera l'URL finale. L'URL est valide pendant 15 minutes à partir du moment où elle est créée. Les informations d'identification de sécurité temporaires et la session de console intégrées dans l'URL sont valides pendant la durée spécifiée dans le paramètre HTTP `SessionDuration` lors de leur demande initiale.

```
https://signin.aws.amazon.com/federation
?Action=login
&Issuer=https%3A%2F%2Fexample.com
&Destination=https%3A%2F%2Fconsole.aws.amazon.com%2F
&SigninToken=VCQgs5qZZt3Q6fn8Tr5EXAMPLEmLnwB7JjUc-SHwnUUWabcRdnWsi4DBn-dvC
CZ85wrD0nmldUcZEXAMPLE-vXYH4Q__mleuF_W2BE5HYexbe9y4Of-kje53SsjNNecATfjIzpW1
WibbnH6YcYRiBoffZBGExbEXAMPLE5aiKX4THWjQKC6gg6alHu6JFrnOJoK3dtP6I9a6hi6yPgm
iOkPZMmNGmhsvVxetKzr8mx3pxhHbMEXAMPLETv1pij0rok3IyCR2YVcIjqwfWv32HU2Xlj471u
3fU6uOfUComeKiqTGX974xzJOZbdmX_t_lLrhEXAMPLEDDIisSnyHGw2xaZZqudm4mo2uTDk9Pv
9l5K0ZCqIgEXAMPLEcA6tgLPykEWGUyH6BdSC6166n4M4JkXIQgac7_7821YqixsNxZ6rsrpzwf
nQoS14O7R0eJCCJ684EXAMPLEZRdBNnuLbUYpz2Iw3vIN0tQgOujwnwydPscM9F7foaEK3jwMkg
Apeb1-6L_OB12MZhuFxx55555EXAMPLEhyETEd4ZulKPdXHkgl6T9ZkIlHz2Uy1RUTUhhUxNtSQ
nWc5xkbBoEcXqpoSIeK7yhje9Vzhd61AEXAMPLElbWeouACEMG6-Vd3dAgFYd6i5FYoyFrZLWvm
0LSG7RyYKeYN5VIzUk3YWQpyjP0RiT5KUrsUi-NEXAMPLExMOMdoODBEgKQsk-iu2ozh6r8bxwC
RNhujg
```

Exemple de code utilisant Python

L'exemple suivant montre comment utiliser Python pour créer par programmation une URL qui donne aux utilisateurs un accès direct à Console de gestion AWS. Voici deux exemples :

- Fédérez via des requêtes GET pour AWS
- Fédérez via des requêtes POST vers AWS

Les deux exemples utilisent l'AssumeRoleAPI AWS SDK pour Python (Boto3) et pour obtenir des informations d'identification de sécurité temporaires.

N'incluez pas `SessionDuration` si vos informations d'identification `AssumeRoleSession` proviennent d'un chaînage de rôles. Si vous incluez `SessionDuration`, l'opération échouera.

Utiliser des requêtes GET

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'

# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your Compte AWS,
# call "AssumeRole" to get temporary access keys for the role or federated principal

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,
# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleSession",
)

# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
json_string_with_temp_credentials = json.dumps(url_credentials)
```

```
# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = "?Action=getSigninToken"
request_parameters += "&SessionDuration=43200"
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)
request_parameters += "&Session=" +
 quote_plus_function(json_string_with_temp_credentials)
request_url = "https://signin.aws.amazon.com/federation" + request_parameters
r = requests.get(request_url)
# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create URL where users can use the sign-in token to sign in to
# the console. This URL must be used within 15 minutes after the
# sign-in token was issued.
request_parameters = "?Action=login"
request_parameters += "&Issuer=Example.org"
request_parameters += "&Destination=" + quote_plus_function("https://
console.aws.amazon.com/")
request_parameters += "&SigninToken=" + signin_token["SigninToken"]
request_url = "https://signin.aws.amazon.com/federation" + request_parameters

# Send final URL to stdout
print (request_url)
```

Utiliser des requêtes POST

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'
import os
from selenium import webdriver # 'pip install selenium', 'brew install chromedriver'

# Step 1: Authenticate user in your own identity system.
```

```
# Step 2: Using the access keys for an IAM user in your A Compte AWS,
# call "AssumeRole" to get temporary access keys for the role or federated principal

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,

# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)

sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleDemoSession",
)

# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = {}
request_parameters['Action'] = 'getSigninToken'
request_parameters['SessionDuration'] = '43200'
```

```
request_parameters['Session'] = json_string_with_temp_credentials

request_url = "https://signin.aws.amazon.com/federation"
r = requests.post( request_url, data=request_parameters)

# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create a POST request where users can use the sign-in token to sign in to
# the console. The POST request must be made within 15 minutes after the
# sign-in token was issued.
request_parameters = {}
request_parameters['Action'] = 'login'
request_parameters['Issuer']='Example.org'
request_parameters['Destination'] = 'https://console.aws.amazon.com/'
request_parameters['SigninToken'] =signin_token['SigninToken']

jsrequest = '''
var form = document.createElement('form');
form.method = 'POST';
form.action = '{request_url}';
request_parameters = {request_parameters}
for (var param in request_parameters) {{
    if (request_parameters.hasOwnProperty(param)) {{
        const hiddenField = document.createElement('input');
        hiddenField.type = 'hidden';
        hiddenField.name = param;
        hiddenField.value = request_parameters[param];
        form.appendChild(hiddenField);
    }}
}}
document.body.appendChild(form);
form.submit();
'''.format(request_url=request_url, request_parameters=request_parameters)

driver = webdriver.Chrome()
driver.execute_script(jsrequest)
input("Press Enter to close the browser window...")
```

Exemple de code utilisant Java

L'exemple suivant montre comment utiliser Java pour créer par programmation une URL qui donne aux utilisateurs un accès direct à Console de gestion AWS. L'extrait de code suivant utilise le kit SDK AWS pour Java.

```
import java.net.URLEncoder;
import java.net.URL;
import java.net.URLConnection;
import java.io.BufferedReader;
import java.io.InputStreamReader;
// Available at http://www.json.org/java/index.html
import org.json.JSONObject;
import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClient;
import com.amazonaws.services.securitytoken.model.Credentials;
import com.amazonaws.services.securitytoken.model.GetFederationTokenRequest;
import com.amazonaws.services.securitytoken.model.GetFederationTokenResult;


/* Calls to AWS STS API operations must be signed using the access key ID
   and secret access key of an IAM user or using existing temporary
   credentials. The credentials should not be embedded in code. For
   this example, the code looks for the credentials in a
   standard configuration file.
*/
AWSCredentials credentials =
  new PropertiesCredentials(
         AwsConsoleApp.class.getResourceAsStream("AwsCredentials.properties"));

AWSSecurityTokenServiceClient stsClient =
  new AWSSecurityTokenServiceClient(credentials);

GetFederationTokenRequest getFederationTokenRequest =
  new GetFederationTokenRequest();
getFederationTokenRequest.setDurationSeconds(1800);
getFederationTokenRequest.setName("UserName");

// A sample policy for accessing Amazon Simple Notification Service (Amazon SNS) in the
 console.
```

```
String policy = "{\"Version\":\"2012-10-17\",       \"Statement\":[{\"Action\":\"sns:*
\"," +
  "\"Effect\":\"Allow\",\"Resource\":\"*\"}]}";

getFederationTokenRequest.setPolicy(policy);

GetFederationTokenResult federationTokenResult =
  stsClient.getFederationToken(getFederationTokenRequest);

Credentials federatedCredentials = federationTokenResult.getCredentials();

// The issuer parameter specifies your internal sign-in
// page, for example https://mysignin.internal.mycompany.com/.
// The console parameter specifies the URL to the destination console of the
// AWS Management Console. This example goes to Amazon SNS.
// The signin parameter is the URL to send the request to.

String issuerURL = "https://mysignin.internal.mycompany.com/";
String consoleURL = "https://console.aws.amazon.com/sns";
String signInURL = "https://signin.aws.amazon.com/federation";

// Create the sign-in token using temporary credentials,
// including the access key ID,  secret access key, and session token.
String sessionJson = String.format(
  "{\"%1$s\":\"%2$s\",\"%3$s\":\"%4$s\",\"%5$s\":\"%6$s\"}",
  "sessionId", federatedCredentials.getAccessKeyId(),
  "sessionKey", federatedCredentials.getSecretAccessKey(),
  "sessionToken", federatedCredentials.getSessionToken());

// Construct the sign-in request with the request sign-in token action, a
// 12-hour console session duration, and the JSON document with temporary
// credentials as parameters.

String getSigninTokenURL = signInURL +
                           "?Action=getSigninToken" +
                           "&DurationSeconds=43200" +
                           "&SessionType=json&Session=" +
                           URLEncoder.encode(sessionJson,"UTF-8");

URL url = new URL(getSigninTokenURL);

// Send the request to the AWS federation endpoint to get the sign-in token
URLConnection conn = url.openConnection ();
```

```
BufferedReader bufferReader = new BufferedReader(new
  InputStreamReader(conn.getInputStream()));
String returnContent = bufferReader.readLine();

String signinToken = new JSONObject(returnContent).getString("SigninToken");

String signinTokenParameter = "&SigninToken=" + URLEncoder.encode(signinToken,"UTF-8");

// The issuer parameter is optional, but recommended. Use it to direct users
// to your sign-in page when their session expires.

String issuerParameter = "&Issuer=" + URLEncoder.encode(issuerURL, "UTF-8");

// Finally, present the completed URL for the AWS console session to the user

String destinationParameter = "&Destination=" + URLEncoder.encode(consoleURL,"UTF-8");
String loginURL = signInURL + "?Action=login" +
                     signinTokenParameter + issuerParameter + destinationParameter;
```

Exemple montrant comment créer l'URL (Ruby)

L'exemple suivant montre comment utiliser Ruby pour créer par programmation une URL qui donne aux utilisateurs un accès direct à Console de gestion AWS. Cet extrait de code utilise le kit SDK AWS pour Ruby.

```
require 'rubygems'
require 'json'
require 'open-uri'
require 'cgi'
require 'aws-sdk'

# Create a new STS instance
#
# Note: Calls to AWS STS API operations must be signed using an access key ID
# and secret access key. The credentials can be in EC2 instance metadata
# or in environment variables and will be automatically discovered by
# the default credentials provider in the AWS Ruby SDK.
sts = Aws::STS::Client.new()

# The following call creates a temporary session that returns
# temporary security credentials and a session token.
# The policy grants permissions to work
# in the AWS SNS console.
```

```
session = sts.get_federation_token({
  duration_seconds: 1800,
  name: "UserName",
  policy: "{\"Version\":\"2012-10-17\",       \"Statement\":{\"Effect\":\"Allow\",
\"Action\":\"sns:*\",\"Resource\":\"*\"}}",
})

# The issuer value is the URL where users are directed (such as
# to your internal sign-in page) when their session expires.
#
# The console value specifies the URL to the destination console.
# This example goes to the Amazon SNS console.
#
# The sign-in value is the URL of the AWS STS federation endpoint.
issuer_url = "https://mysignin.internal.mycompany.com/"
console_url = "https://console.aws.amazon.com/sns"
signin_url = "https://signin.aws.amazon.com/federation"

# Create a block of JSON that contains the temporary credentials
# (including the access key ID, secret access key, and session token).
session_json = {
  :sessionId => session.credentials[:access_key_id],
  :sessionKey => session.credentials[:secret_access_key],
  :sessionToken => session.credentials[:session_token]
}.to_json

# Call the federation endpoint, passing the parameters
# created earlier and the session information as a JSON block.
# The request returns a sign-in token that's valid for 15 minutes.
# Signing in to the console with the token creates a session
# that is valid for 12 hours.
get_signin_token_url = signin_url +
                       "?Action=getSigninToken" +
                       "&SessionType=json&Session=" +
                       CGI.escape(session_json)

returned_content = URI.parse(get_signin_token_url).read

# Extract the sign-in token from the information returned
# by the federation endpoint.
signin_token = JSON.parse(returned_content)['SigninToken']
signin_token_param = "&SigninToken=" + CGI.escape(signin_token)
```

```
# Create the URL to give to the user, which includes the
# sign-in token and the URL of the console to open.
# The "issuer" parameter is optional but recommended.
issuer_param = "&Issuer=" + CGI.escape(issuer_url)
destination_param = "&Destination=" + CGI.escape(console_url)
login_url = signin_url + "?Action=login" + signin_token_param +
  issuer_param + destination_param
```

Tags pour Gestion des identités et des accès AWS resources

Une balise est un attribut personnalisé que vous pouvez attribue à une ressource AWS . Chaque balise se compose de deux parties :

- Une clé de balise (par exemple, `CostCenter`, `Environment`, `Project` ou `Purpose`).
- Un champ facultatif appelé valeur de balise (par exemple, `111122223333`, `Production` ou le nom d'une équipe). Omettre la valeur de balise équivaut à l'utilisation d'une chaîne vide.

Ensemble, ces éléments sont connus sous le nom de paires clé-valeur. Pour connaître les limites de nombre de balises possibles pour les ressources IAM, veuillez consulter IAM et AWS STS quotas.

> **Note**
>
> Pour plus d'informations sur la sensibilité à la casse des clés de balises et les valeurs de clés de balises, consultez Case sensitivity (français non garanti).

Les balises vous aident à identifier et à organiser vos AWS ressources. De nombreux AWS services prennent en charge le balisage. Vous pouvez donc attribuer le même tag à des ressources provenant de différents services pour indiquer que les ressources sont liées. Par exemple, vous pouvez attribuer la même balise à un rôle IAM que vous affectez à un compartiment Amazon S3. Pour plus d'informations sur les stratégies de balisage, consultez le Guide de l'utilisateur AWS des ressources de balisage.

Outre l'identification, l'organisation et le suivi de vos ressources IAM avec des balises, vous pouvez utiliser des balises dans les politiques IAM afin de contrôler qui peut consulter et interagir avec vos ressources. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Vous pouvez également utiliser des balises AWS STS pour ajouter des attributs personnalisés lorsque vous assumez un rôle ou que vous fédérez un utilisateur. Pour de plus amples informations, veuillez consulter Transmettre les balises de session AWS STS.

Rubriques

- Choisissez un AWS convention de dénomination des balises
- Règles de balisage dans IAM et AWS STS
- Balisage des utilisateurs IAM
- Blisage de rôles IAM
- Balisage des politiques gérées par le client
- Balisage de fournisseurs d'identité OpenID Connect (OIDC)
- Balisage de fournisseurs d'identité SAML IAM
- Balisage de profils d'instance pour les rôles Amazon EC2
- Balisage des certificats de serveur
- Balisage des dispositifs MFA virtuels
- Marquer les instances du gestionnaire d'accès aux comptes
- Transmettre les balises de session AWS STS

Choisissez un AWS convention de dénomination des balises

Lorsque vous commencez à attacher des balises à vos ressources IAM, choisissez soigneusement votre convention de dénomination de balises. Appliquez la même convention à tous vos AWS tags. Cela est particulièrement important si vous utilisez des balises dans les politiques pour contrôler l'accès aux AWS ressources. Si vous utilisez déjà des balises dans AWS, passez en revue votre convention de dénomination et ajustez-la en conséquence.

> **Note**
>
> Si votre compte est membre de AWS Organizations, consultez les politiques relatives aux balises dans le guide de l' AWS Organizations utilisateur pour en savoir plus sur l'utilisation des balises dans AWS Organizations.

Bonnes pratiques pour nommer les balises

Voici quelques bonnes pratiques et conventions pour la dénomination des balises.

Veillez à ce que les noms de balises soient utilisés de manière cohérente. Par exemple, les balises `CostCenter` et `costcenter` sont différentes, de sorte que l'une peut être configurée en tant que balise de répartition des coûts pour l'analyse financière et la confection de rapports, alors que l'autre peut ne pas l'être. De même, la `Name` balise apparaît dans la AWS console pour de nombreuses ressources, mais pas la `name` balise. Pour plus d'informations sur la sensibilité à la casse des clés de balises et les valeurs de clés de balises, consultez Case sensitivity (français non garanti).

Un certain nombre de balises sont prédéfinies AWS ou créées automatiquement par différents AWS services. De nombreux noms de balises AWS définis utilisent uniquement des minuscules, des tirets séparant les mots du nom et des préfixes pour identifier le service source de la balise. Par exemple :

- `aws:ec2spot:fleet-request-id` identifie la demande d'instance Spot Amazon EC2 ponctuelle qui a lancé l'instance.
- `aws:cloudformation:stack-name`identifie la CloudFormation pile qui a créé la ressource.
- `elasticbeanstalk:environment-name` identifie l'application qui a créé la ressource.

Pensez à nommer vos balises en minuscules, avec des traits d'union séparant les mots et un préfixe identifiant le nom de l'organisation ou le nom abrégé. Par exemple, pour une société fictive nommée AnyCompany, vous pouvez définir des balises telles que :

- `anycompany:cost-center` pour identifier le code du centre de coûts interne
- `anycompany:environment-type` pour déterminer si l'environnement est un environnement de développement, de test ou de production
- `anycompany:application-id` pour identifier l'application pour laquelle la ressource a été créée

Le préfixe garantit que les balises sont clairement identifiées comme ayant été définies par votre organisation et AWS non par un outil tiers que vous pourriez utiliser. L'utilisation de minuscules et de traits d'union pour les séparateurs évite toute confusion quant à l'utilisation de majuscules pour le nom d'une balise. Par exemple, `anycompany:project-id`est plus simple à mémoriser que `ANYCOMPANY:ProjectID`, `anycompany:projectID` ou`Anycompany:ProjectId`.

Règles de balisage dans IAM et AWS STS

Un certain nombre de conventions régissent la création et l'application des balises dans IAM et AWS STS.

Dénomination des balises

Respectez les conventions suivantes lors de la formulation d'une convention de dénomination de balises pour les ressources IAM, les sessions d'attribution de AWS STS rôles et AWS STS les sessions utilisateur fédérées :

Character requirements (Exigences relatives aux caractères) : les clés et valeurs des balises peuvent inclure n'importe quelle combinaison de lettres, chiffres, espaces et caractères _ . : / = + - @.

Sensibilité à la casse : la sensibilité à la casse pour les clés de balise diffère selon le type de ressource IAM qui est balisé. Les valeurs clés de balise pour les utilisateurs et les rôles IAM ne sont pas sensibles à la casse, mais la casse est conservée. Cela signifie que vous ne pouvez pas avoir des clés de balise **`Department`** et **`department`** distinctes. Si vous avez balisé un utilisateur avec la balise **`Department=finance`** et que vous ajoutez la balise **`department=hr`**, elle remplace la première balise. Une deuxième balise n'est pas ajoutée.

Pour les autres types de ressources IAM, les valeurs de clé de balise sont sensibles à la casse. Cela signifie que vous pouvez avoir des clés de balise **`Costcenter`** et **`costcenter`** séparées. Par exemple, si vous avez balisé une politique gérée par le client avec la balise **`Costcenter = 1234`** et que vous ajoutez la balise **`costcenter = 5678`**, la politique aura à la fois les clés de balise **`Costcenter`** et **`costcenter`**.

À titre de bonne pratique, nous vous recommandons d'éviter d'utiliser des balises similaires avec un traitement de casse incohérent. Nous vous recommandons de choisir une stratégie pour l'utilisation de majuscules pour le nom d'une balise et de mettre en œuvre cette stratégie de manière cohérente sur tous les types de ressources. Pour en savoir plus sur les meilleures pratiques en matière de balisage, consultez la section AWS Ressources de balisage dans le. Références générales AWS

Les listes suivantes présentent les différences de sensibilité à la casse pour les clés de balise attachées aux ressources IAM.

Les valeurs de clé de balise ne sont pas sensibles à la casse :

- Rôles IAM
- Utilisateurs IAM

Les valeurs de clé de balise sont sensibles à la casse :

- Politiques gérées par le client
- Profils d'instance
- Fournisseurs d'identité OpenID Connect
- Fournisseurs d'identité SAML
- Certificats de serveur
- Appareils MFA virtuels

En outre, les règles suivantes s'appliquent :

- Vous ne pouvez pas créer une clé ou valeur de balise qui commence par le texte **`aws:`**. Ce préfixe de balise est réservé à un usage AWS interne.
- Vous pouvez créer une balise avec une valeur vide comme **`phoneNumber =`** . Vous ne pouvez pas créer une clé de balise vide.
- Vous ne pouvez pas spécifier plusieurs valeurs dans une seule balise, mais vous pouvez créer une structure multivaleur personnalisée dans la valeur unique. Par exemple, supposons que l'utilisateur Zhang fonctionne sur l'équipe d'ingénierie et l'équipe d'assurance qualité. Si vous attachez la balise **`team = Engineering`**, puis attachez la balise **`team = QA`**, vous modifiez la valeur de la balise de **`Engineering`** en **`QA`**. Au lieu de cela, vous pouvez inclure plusieurs valeurs dans une seule balise avec un séparateur personnalisé. Dans cet exemple, vous pouvez attacher la balise **`team = Engineering:QA`** à Zhang.

> **Note**
>
> Pour contrôler l'accès aux ingénieurs de cet exemple à l'aide de la balise **`team`**, vous devez créer une politique qui autorise chaque configuration susceptible d'inclure **`Engineering`**, y compris **`Engineering:QA`**. Pour en savoir plus sur l'utilisation des balises dans les politiques consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Application et modification de balises

Observez les conventions suivantes lorsque vous attachez des balises aux ressources IAM :

- Vous pouvez baliser la plupart des ressources IAM, mais pas les groupes, les rôles endossés, les rapports d'accès, les dispositifs MFA matériels.
- Vous ne pouvez pas utiliser Tag Editor pour baliser les ressources IAM. Tag Editor ne prend pas en charge les balises IAM. Pour plus d'informations sur l'utilisation de Tag Editor avec d'autres services, consultez Utilisation de Tag Editor dans le Guide de l'utilisateur Groupes de ressources AWS .
- Vous devez disposer d'autorisations spécifiques pour baliser une ressource IAM. Pour ajouter ou supprimer des balises aux ressources, vous devez également disposer d'une autorisation pour afficher la liste des balises. Pour plus d'informations, consultez la liste des rubriques pour chaque ressource IAM à la fin de cette page.
- Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.
- Vous pouvez appliquer la même balise à plusieurs ressources IAM. Par exemple, supposons que vous ayez un service nommé `AWS_Development` comprenant 12 membres. Vous pouvez avoir 12 utilisateurs et un rôle avec la clé de balise **`department`** et la valeur **`awsDevelopment`** (**`department = awsDevelopment`**). Vous pouvez également utiliser la même balise sur les ressources d'autres services qui prennent en charge le balisage.
- Les entités IAM (utilisateurs ou rôles) ne peuvent pas avoir plusieurs instances de la même clé de balise. Par exemple, si vous avez un utilisateur avec la paire clé-valeur de balise **`costCenter = 1234`**, vous pouvez ensuite attacher la paire clé-valeur de balise **`costCenter = 5678`**. IAM met à jour la valeur de la balise **`costCenter`** à **`5678`**.
- Pour modifier une balise qui est attachée à une entité IAM (utilisateur ou rôle), attachez une balise avec une nouvelle valeur pour remplacer la balise existante. Par exemple, supposons que vous disposiez d'un utilisateur avec la paire clé-valeur de balise **`department = Engineering`**. Si vous devez déplacer l'utilisateur vers le service d'assurance qualité, vous pouvez attacher la paire clé-valeur de balise **`department = QA`** à l'utilisateur. Il s'ensuit que la valeur **`Engineering`** de la clé de balise **`department`** est remplacée par la valeur **`QA`**.

Balisage des utilisateurs IAM

Vous pouvez utiliser des paires clé-valeur de balise IAM pour ajouter des attributs personnalisés à un utilisateur IAM. Par exemple, pour ajouter les informations relatives à un utilisateur, vous pouvez ajouter la clé de balise **`location`** et la valeur de balise **`us_wa_seattle`**. Ou vous pouvez utiliser trois paires clé-valeur de balise aux emplacements distincts : **`loc-country = us`**, **`loc-state = wa`** et **`loc-city = seattle`**. Vous pouvez utiliser les balises pour contrôler l'accès d'un utilisateur

aux ressources d'une entité ou pour contrôler les balises qui peuvent être attachées à un utilisateur. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Vous pouvez également utiliser des balises AWS STS pour ajouter des attributs personnalisés lorsque vous assumez un rôle ou que vous fédérez un utilisateur. Pour de plus amples informations, veuillez consulter Transmettre les balises de session AWS STS.

Autorisations requises pour le balisage des utilisateurs IAM

Vous devez configurer les autorisations pour permettre à un utilisateur IAM de baliser d'autres utilisateurs. Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListUserTags`
- `iam:TagUser`
- `iam:UntagUser`

Pour autoriser un utilisateur IAM à ajouter, afficher la liste ou supprimer une balise pour un utilisateur spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'utilisateur IAM qui doit gérer les balises. Utilisez votre numéro de compte et *<username>* remplacez-le par le nom de l'utilisateur dont les tags doivent être gérés. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Pour autoriser un utilisateur IAM à gérer lui-même les balises

Ajoutez l'instruction suivante à la politique d'autorisations pour les utilisateurs qui autorisent les utilisateurs à gérer leurs propres balises. Pour apprendre à créer une politique à l'aide de cet

exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::user/${aws:username}"
}
```

Pour autoriser un utilisateur IAM à ajouter une balise à un utilisateur spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'utilisateur IAM qui doit ajouter, mais non pas supprimer, les balises d'un utilisateur spécifique.

> **Note**
>
> L'action `iam:TagUser` nécessite d'inclure également l'action `iam:ListUserTags`.

Pour utiliser cette politique, *<username>* remplacez-la par le nom de l'utilisateur dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Vous pouvez également utiliser une politique AWS gérée, par exemple IAMFullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les utilisateurs IAM (console)

Vous pouvez gérer les balises pour les utilisateurs IAM depuis la Console de gestion AWS.

Pour gérer les balises sur les utilisateurs (console)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation de la console, choisissez Users (Utilisateurs), puis choisissez le nom de l'utilisateur que vous souhaitez modifier.
3. Choisissez l'onglet Balises, puis effectuez l'une des actions suivantes :
   - Choisissez Ajouter une nouvelle balise si l'utilisateur ne dispose pas encore de balises.
   - Choisissez Manage tags (Gérer les balises) pour gérer l'ensemble de balises existant.
4. Ajoutez ou supprimez des balises pour terminer l'ensemble de balises. Ensuite, choisissez Enregistrer les modifications.

Gestion des balises sur les utilisateurs IAM (AWS CLI or AWS API)

Vous pouvez afficher la liste, attacher ou supprimer des balises pour les utilisateurs IAM. Vous pouvez utiliser l' AWS API AWS CLI ou pour gérer les balises pour les utilisateurs IAM.

Pour répertorier les balises actuellement associées à un utilisateur IAM (AWS CLI or AWS API)

- AWS CLI: aws iam list-user-tags
- AWS API : ListUserTags

Pour associer des balises à un utilisateur IAM (AWS CLI or AWS API)

- AWS CLI: aws iam tag-user
- AWS API : TagUser

Pour supprimer des balises d'un utilisateur IAM (AWS CLI or AWS API)

- AWS CLI: aws iam untag-user
- AWS API : UntagUser

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Blisage de rôles IAM

Vous pouvez utiliser des paires clé-valeur de balises pour ajouter des attributs personnalisés à un rôle IAM. Par exemple, pour ajouter les informations relatives à un rôle, vous pouvez ajouter la clé de balise **`location`** et la valeur de balise **`us_wa_seattle`**. Ou vous pouvez utiliser trois paires clé-valeur de balise aux emplacements distincts : **`loc-country = us`**, **`loc-state = wa`** et **`loc-city = seattle`**. Vous pouvez utiliser les balises pour contrôler l'accès d'un rôle aux ressources d'une entité ou pour contrôler les balises qui peuvent être attachées à un rôle. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Vous pouvez également utiliser des balises AWS STS pour ajouter des attributs personnalisés lorsque vous assumez un rôle ou que vous fédérez un utilisateur. Pour de plus amples informations, veuillez consulter Transmettre les balises de session AWS STS.

Autorisations requises pour le balisage des rôles IAM

Vous devez configurer les autorisations de manière à permettre à un rôle IAM de baliser d'autres entités (utilisateurs ou rôles). Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListRoleTags`
- `iam:TagRole`
- `iam:UntagRole`
- `iam:ListUserTags`
- `iam:TagUser`
- `iam:UntagUser`

Pour autoriser un rôle IAM à ajouter, afficher la liste ou supprimer une balise pour un utilisateur spécifique

Ajoutez l'instruction suivante à la politique d'autorisations du rôle IAM qui doit gérer les balises. Utilisez votre numéro de compte et *<username>* remplacez-le par le nom de l'utilisateur dont les tags doivent être gérés. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Pour autoriser un rôle IAM à ajouter une balise à un utilisateur spécifique

Ajoutez l'instruction suivante à la politique d'autorisations du rôle IAM qui doit ajouter, mais non pas supprimer, les balises d'un utilisateur spécifique.

Pour utiliser cette politique, *<username>* remplacez-la par le nom de l'utilisateur dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Pour autoriser un rôle IAM à ajouter, afficher la liste ou supprimer une balise pour un rôle spécifique

Ajoutez l'instruction suivante à la politique d'autorisations du rôle IAM qui doit gérer les balises. Remplacez *<rolename>* par le nom du rôle dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
```

```
    "Effect": "Allow",
    "Action": [
        "iam:ListRoleTags",
        "iam:TagRole",
        "iam:UntagRole"
    ],
    "Resource": "arn:aws:iam::<account-number>:role/<rolename>"
}
```

Vous pouvez également utiliser une politique AWS gérée, par exemple IAMFullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les rôles IAM (console)

Vous pouvez gérer les balises pour les rôles IAM depuis la Console de gestion AWS.

Pour gérer les balises sur les rôles (console)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation de la console, choisissez Roles (Rôles), puis choisissez le nom du rôle que vous souhaitez modifier.
3. Choisissez l'onglet Balises, puis effectuez l'une des actions suivantes :
   - Choisissez Add new tag (Ajouter une nouvelle balise) si le rôle ne dispose pas encore de balises.
   - Choisissez Manage tags (Gérer les balises) pour gérer l'ensemble de balises existant.
4. Ajoutez ou supprimez des balises pour terminer l'ensemble de balises. Ensuite, choisissez Save changes (Enregistrer les modifications).

Gestion des balises sur les rôles IAM (AWS CLI or AWS API)

Vous pouvez afficher la liste, attacher ou supprimer des balises pour les rôles IAM. Vous pouvez utiliser l' AWS API AWS CLI ou pour gérer les balises des rôles IAM.

Pour répertorier les balises actuellement associées à un rôle IAM (AWS CLI or AWS API)

- AWS CLI: aws iam list-role-tags
- AWS API : ListRoleTags

Pour associer des balises à un rôle IAM (AWS CLI or AWS API)

- AWS CLI: aws iam tag-role
- AWS API : TagRole

Pour supprimer des balises d'un rôle IAM (AWS CLI or AWS API)

- AWS CLI: aws iam untag-role
- AWS API : UntagRole

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Balisage des politiques gérées par le client

Vous pouvez utiliser des paires clé-valeur de balises IAM pour ajouter des attributs personnalisés à vos politiques gérées par le client. Par exemple, pour baliser une politique avec des informations de service, vous pouvez ajouter la clé de balise **`Department`** et la valeur de balise **`eng`**. Ou bien, vous pouvez baliser les politiques pour indiquer qu'elles sont destinées à un environnement spécifique, comme **`Environment = lab`**. Vous pouvez utiliser les balises pour contrôler l'accès aux ressources ou pour contrôler les balises qui peuvent être attachées à une ressource. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Vous pouvez également utiliser des balises AWS STS pour ajouter des attributs personnalisés lorsque vous assumez un rôle ou que vous fédérez un utilisateur. Pour de plus amples informations, veuillez consulter Transmettre les balises de session AWS STS.

Autorisations requises pour baliser les politiques gérées par le client

Vous devez configurer des autorisations pour autoriser une entité IAM (utilisateurs ou rôles) à baliser les politiques gérées par le client. Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListPolicyTags`

- `iam:TagPolicy`
- `iam:UntagPolicy`

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter, afficher la liste ou supprimer une balise pour une politique gérée par le client

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit gérer les balises. Utilisez votre numéro de compte et *<policyname>* remplacez-le par le nom de la politique dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy",
        "iam:UntagPolicy"
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter une balise à une politique gérée par le client spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit ajouter, mais non pas supprimer, les balises d'une politique spécifique.

> **Note**
>
> L'action `iam:TagPolicy` nécessite d'inclure également l'action `iam:ListPolicyTags`.

Pour utiliser cette politique, *<policyname>* remplacez-la par le nom de la stratégie dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
```

```
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy"
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

Vous pouvez également utiliser une politique AWS gérée, par exemple IAMFullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les politiques gérées par le client IAM (console)

Vous pouvez gérer les balises pour les politiques gérées par le client IAM à partir de la Console de gestion AWS.

Pour gérer les balises sur les politiques gérées par le client (console)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation de la console, choisissez Policies (Politiques), puis choisissez le nom de la politique gérée par le client que vous souhaitez modifier.
3. Choisissez l'onglet Balises, puis Gérer les balises.
4. Ajoutez ou supprimez des balises pour terminer l'ensemble de balises. Ensuite, choisissez Enregistrer les modifications.

Gestion des balises sur les politiques gérées par les clients IAM (AWS CLI or AWS API)

Vous pouvez afficher la liste, attacher ou supprimer des balises pour les politiques gérées par le client IAM. Vous pouvez utiliser l' AWS API AWS CLI ou pour gérer les balises pour les politiques IAM gérées par les clients.

Pour répertorier les balises actuellement associées à une politique IAM gérée par le client (AWS CLI or AWS API)

- AWS CLI : aws iam list-policy-tags
- AWS API : ListPolicyTags

Pour associer des balises à une politique IAM gérée par le client (AWS CLI or AWS API)

- AWS CLI : aws iam tag-policy
- AWS API : TagPolicy

Pour supprimer des balises d'une politique gérée par le client IAM (AWS CLI or AWS API)

- AWS CLI : aws iam untag-policy
- AWS API : UntagPolicy

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Balisage de fournisseurs d'identité OpenID Connect (OIDC)

Vous pouvez utiliser des paires clé-valeur de balises IAM pour ajouter des attributs personnalisés aux fournisseurs d'identité OpenID Connect (OIDC) IAM. Par exemple, pour identifier un fournisseur d'identité OIDC, vous pouvez ajouter la clé de balise **`google`** et la valeur de balise **`oidc`**. Vous pouvez utiliser les balises pour contrôler l'accès aux ressources ou pour contrôler les balises qui peuvent être attachées à un objet. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Autorisations requises pour baliser les fournisseurs d'identité OIDC IAM

Vous devez configurer les autorisations de manière à permettre à une entité IAM (utilisateur ou rôle) de baliser des fournisseurs d'identité OIDC IAM. Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListOpenIDConnectProviderTags`
- `iam:TagOpenIDConnectProvider`
- `iam:UntagOpenIDConnectProvider`

Pour autoriser une entité IAM à ajouter, afficher la liste ou supprimer une balise pour un fournisseur d'identité OIDC IAM

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit gérer les balises. Utilisez votre numéro de compte et *<OIDCProviderName>* remplacez-le par le nom du fournisseur OIDC dont les tags doivent être gérés. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider",
        "iam:UntagOpenIDConnectProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
}
```

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter une balise à un fournisseur d'identité OIDC IAM spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit ajouter, mais non pas supprimer, les balises d'un fournisseur d'identité spécifique.

> **Note**
>
> L'action `iam:TagOpenIDConnectProvider` nécessite d'inclure également l'action `iam:ListOpenIDConnectProviderTags`.

Pour utiliser cette politique, remplacez *<OIDCProviderName>* par le nom du fournisseur OIDC dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
```

```
}
```

Vous pouvez également utiliser une politique AWS gérée, par exemple IAMFullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les fournisseurs d'identité OIDC IAM (console)

Vous pouvez gérer les balises pour les fournisseurs d'identité OIDC IAM à partir de la Console de gestion AWS.

Pour gérer les balises sur les fournisseurs d'identité OIDC (console)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation de la console, sélectionnez Identity providers (Fournisseurs d'identité), puis le nom du fournisseur d'identité que vous souhaitez modifier.
3. Choisissez l'onglet Balises, puis dans la section Balises, choisissez Gérer les balises, puis effectuez l'une des actions suivantes :
   - Choisissez Add tag (Ajouter une balise) si le fournisseur d'identité OIDC n'a pas encore de balises ou pour ajouter une nouvelle balise.
   - Modifiez les clés et les valeurs de balise existantes.
   - Choisissez Remove tag (Supprimer une balise) pour supprimer une balise.
4. Ensuite, choisissez Enregistrer les modifications.

Gestion des balises sur les fournisseurs d'identité IAM OIDC (AWS CLI or AWS API)

Vous pouvez afficher la liste, attacher ou supprimer des balises pour les fournisseurs d'identité OIDC IAM. Vous pouvez utiliser l' AWS API AWS CLI ou pour gérer les balises des fournisseurs d'identité IAM OIDC.

Pour répertorier les balises actuellement associées à un fournisseur d'identité IAM OIDC (AWS CLI or AWS API)

- AWS CLI : aws iam list-open-id-connect-provider-tags
- AWS API : ListOpenIDConnectProviderTags

Pour associer des balises à un fournisseur d'identité IAM OIDC (AWS CLI or AWS API)

- AWS CLI : aws iam tag-open-id-connect-provider
- AWS API : TagOpenIDConnectProvider

Pour supprimer des balises d'un fournisseur d'identité IAM OIDC (AWS CLI or AWS API)

- AWS CLI : aws iam untag-open-id-connect-provider
- AWS API : UntagOpenIDConnectProvider

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Balisage de fournisseurs d'identité SAML IAM

Vous pouvez utiliser des paires clé-valeur de balises IAM pour ajouter des attributs personnalisés aux fournisseurs d'identité SAML. Par exemple, pour identifier un fournisseur, vous pouvez ajouter la clé de balise **`okta`** et la valeur de balise **`saml`**. Vous pouvez utiliser les balises pour contrôler l'accès aux ressources ou pour contrôler les balises qui peuvent être attachées à un objet. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Autorisations requises pour baliser les fournisseurs d'identité SAML

Vous devez configurer les autorisations pour autoriser une entité IAM (utilisateurs ou rôles) à baliser les fournisseurs d'identité basés sur SAML 2.0 (). IdPs Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListSAMLProviderTags`
- `iam:TagSAMLProvider`
- `iam:UntagSAMLProvider`

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter, afficher la liste ou supprimer une balise pour un fournisseur d'identité SAML

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit gérer les balises. Utilisez votre numéro de compte et *<SAMLProviderName>* remplacez-le par le nom du fournisseur SAML dont les tags doivent être gérés. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
        "iam:TagSAMLProvider",
        "iam:UntagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
}
```

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter une balise à un fournisseur d'identité SAML spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit ajouter, mais non pas supprimer, les balises d'un fournisseur d'identité SAML spécifique.

> **Note**
>
> L'action `iam:TagSAMLProvider` nécessite d'inclure également l'action `iam:ListSAMLProviderTags`.

Pour utiliser cette politique, remplacez *<SAMLProviderName>* par le nom du fournisseur SAML dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
        "iam:TagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
```

```
}
```

Vous pouvez également utiliser une politique AWS gérée, par exemple IAMFullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les fournisseurs d'identité SAML IAM (console)

Vous pouvez gérer les balises pour les fournisseurs d'identité SAML IAM à partir de la Console de gestion AWS.

Pour gérer les balises sur les fournisseurs d'identité SAML (console)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation de la console, sélectionnez Identity providers (Fournisseurs d'identité), puis le nom du fournisseur d'identité SAML que vous souhaitez modifier.
3. Choisissez l'onglet Balises, puis dans la section Balises, choisissez Gérer les balises, puis effectuez l'une des actions suivantes :
   - Choisissez Add tag (Ajouter une balise) si le fournisseur d'identité SAML n'a pas encore de balises ou pour ajouter une nouvelle balise.
   - Modifiez les clés et les valeurs de balise existantes.
   - Choisissez Remove tag (Supprimer une balise) pour supprimer une balise.
4. Ajoutez ou supprimez des balises pour terminer l'ensemble de balises. Ensuite, choisissez Enregistrer les modifications.

Gestion des balises sur les fournisseurs d'identité IAM SAML (AWS CLI or AWS API)

Vous pouvez afficher la liste, attacher ou supprimer des balises pour les fournisseurs d'identité SAML IAM. Vous pouvez utiliser l' AWS API AWS CLI ou pour gérer les balises des fournisseurs d'identité IAM SAML.

Pour répertorier les balises actuellement associées à un fournisseur d'identité SAML (AWS CLI or AWS API)

- AWS CLI : aws iam list-saml-provider-tags
- AWS API : ListSAMLProviderTags

Pour associer des balises à un fournisseur d'identité SAML (AWS CLI or AWS API)

- AWS CLI : aws iam tag-saml-fournisseur
- AWS API : TagSamlProvider

Pour supprimer des balises d'un fournisseur d'identité SAML (AWS CLI or AWS API)

- AWS CLI : aws iam untag-saml-fournisseur
- AWS API : UntagSamlProvider

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Balisage de profils d'instance pour les rôles Amazon EC2

Lorsque vous lancez une instance Amazon EC2, vous spécifiez un rôle IAM à associer à celle-ci. Un profil d'instance est un conteneur pour un rôle IAM que vous pouvez utiliser pour transmettre les informations liées au rôle à une instance Amazon EC2 lorsque l'instance démarre. Vous pouvez baliser les profils d'instance lorsque vous utilisez l' AWS API AWS CLI or.

Vous pouvez utiliser des paires clé-valeur de balise IAM pour ajouter des attributs personnalisés à un profil d'instance. Par exemple, pour ajouter des informations de service à un profil d'instance, vous pouvez ajouter la clé de balise **`access-team`** et la valeur de balise **`eng`**. Cela donne aux principaux avec des balises correspondantes l'accès aux profils d'instance avec la même balise. Vous pouvez utiliser plusieurs paires clé-valeur de balise pour spécifier une équipe et un projet : **`access-team = eng`** , et **`project = peg`**. Vous pouvez utiliser les balises pour contrôler l'accès d'un utilisateur aux ressources d'une entité ou pour contrôler les balises qui peuvent être attachées à un utilisateur. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Vous pouvez également utiliser des balises AWS STS pour ajouter des attributs personnalisés lorsque vous assumez un rôle ou que vous fédérez un utilisateur. Pour de plus amples informations, veuillez consulter Transmettre les balises de session AWS STS.

Autorisations requises pour le balisage des profils d'instance

Vous devez configurer les autorisations de manière à permettre à une entité IAM (utilisateur ou rôle) de baliser des profils d'instance. Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListInstanceProfileTags`
- `iam:TagInstanceProfile`
- `iam:UntagInstanceProfile`

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter, afficher la liste ou supprimer une balise pour un profil d'instance

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit gérer les balises. Utilisez votre numéro de compte et *<InstanceProfileName>* remplacez-le par le nom du profil d'instance dont les tags doivent être gérés. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListInstanceProfileTags",
        "iam:TagInstanceProfile",
        "iam:UntagInstanceProfile"
    ],
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
}
```

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter une balise à un profil d'instance spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit ajouter, mais non pas supprimer, les balises d'un profil d'instance spécifique.

> **Note**
>
> L'action `iam:TagInstanceProfile` nécessite d'inclure également l'action `iam:ListInstanceProfileTags`.

Pour utiliser cette politique, remplacez *<InstanceProfileName>* par le nom du profil d'instance dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListInstanceProfileTags",
        "iam:TagInstanceProfile"
    ],
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
}
```

Vous pouvez également utiliser une politique AWS gérée, par exemple IAMFullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les profils d'instance (AWS CLI or AWS API)

Vous pouvez afficher la liste, attacher ou supprimer des balises pour des profils d'instance. Vous pouvez utiliser l' AWS API AWS CLI ou pour gérer les balises des profils d'instance.

Pour répertorier les balises actuellement associées à un profil d'instance (AWS CLI or AWS API)

- AWS CLI : aws iam list-instance-profile-tags
- AWS API : ListInstanceProfileTags

Pour associer des balises à un profil d'instance (AWS CLI or AWS API)

- AWS CLI : aws iam tag-instance-profile
- AWS API : TagInstanceProfile

Pour supprimer des balises d'un profil d'instance (AWS CLI or AWS API)

- AWS CLI : aws iam untag-instance-profile
- AWS API : UntagInstanceProfile

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Balisage des certificats de serveur

Si vous utilisez IAM pour gérer les SSL/TLS certificats, vous pouvez baliser les certificats de serveur dans IAM à l'aide de l'API AWS CLI or AWS . Pour les certificats d'une région prise en charge par AWS Certificate Manager (ACM), nous vous recommandons d'utiliser ACM au lieu d'IAM pour provisionner, gérer et déployer vos certificats de serveur. Dans les régions non prises en charge, vous devez utiliser IAM en tant que gestionnaire de certificats. Pour savoir quelles régions sont prises en charge par ACM, consultez Points de terminaison et quotas AWS Certificate Manager (français non garanti) dans Références générales AWS.

Vous pouvez utiliser des paires clé-valeur de balise IAM pour ajouter des attributs personnalisés à un certificat de serveur. Par exemple, pour ajouter des informations sur le propriétaire ou l'administrateur d'un certificat de serveur, ajoutez la clé de balise **`owner`** et la valeur de balise **`net-eng`**. Vous pouvez également spécifier un centre de coûts en ajoutant la clé de balise **`CostCenter`** et la valeur de balise **`1234`**. Vous pouvez utiliser les balises pour contrôler l'accès aux ressources ou pour contrôler les balises qui peuvent être attachées à des ressources. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Vous pouvez également utiliser des balises AWS STS pour ajouter des attributs personnalisés lorsque vous assumez un rôle ou que vous fédérez un utilisateur. Pour de plus amples informations, veuillez consulter Transmettre les balises de session AWS STS.

Autorisations requises pour le balisage des certificats de serveur

Vous devez configurer les autorisations de manière à permettre à une entité IAM (utilisateur ou rôle) de baliser des certificats de serveur. Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListServerCertificateTags`
- `iam:TagServerCertificate`
- `iam:UntagServerCertificate`

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter, afficher la liste ou supprimer une balise pour un certificat de serveur

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit gérer les balises. Utilisez votre numéro de compte et *<CertificateName>* remplacez-le par le nom du certificat de serveur dont les tags doivent être gérés. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate",
        "iam:UntagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
}
```

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter une balise à un certificat de serveur spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit ajouter, mais non pas supprimer, les balises d'un certificat de serveur spécifique.

> **Note**
>
> L'action `iam:TagServerCertificate` nécessite d'inclure également l'action `iam:ListServerCertificateTags`.

Pour utiliser cette politique, remplacez *<CertificateName>* par le nom du certificat de serveur dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
```

```
}
```

Vous pouvez également utiliser une politique AWS gérée, par exemple IAMFullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les certificats de serveur (AWS CLI or AWS API)

Vous pouvez afficher la liste, attacher ou supprimer des balises pour les certificats de serveur. Vous pouvez utiliser l' AWS API AWS CLI ou pour gérer les balises des certificats de serveur.

Pour répertorier les balises actuellement associées à un certificat de serveur (AWS CLI or AWS API)

- AWS CLI : aws iam list-server-certificat-tags
- AWS API : ListServerCertificateTags

Pour associer des balises à un certificat de serveur (AWS CLI or AWS API)

- AWS CLI : aws iam tag-server-certificat
- AWS API : TagServerCertificate

Pour supprimer des balises d'un certificat de serveur (AWS CLI or AWS API)

- AWS CLI : aws iam untag-server-certificat
- AWS API : UntagServerCertificate

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Balisage des dispositifs MFA virtuels

Vous pouvez utiliser des paires clé-valeur de balise IAM pour ajouter des attributs personnalisés à un appareil MFA virtuel. Par exemple, pour ajouter des informations de centre de coûts pour l'appareil MFA virtuel d'un utilisateur, vous pouvez ajouter la clé de balise **`CostCenter`** et la valeur de balise **`1234`**. Vous pouvez utiliser les balises pour contrôler l'accès aux ressources ou pour

contrôler les balises qui peuvent être attachées à un objet. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Vous pouvez également utiliser des balises AWS STS pour ajouter des attributs personnalisés lorsque vous assumez un rôle ou que vous fédérez un utilisateur. Pour de plus amples informations, veuillez consulter Transmettre les balises de session AWS STS.

Autorisations requises pour le balisage des appareils MFA virtuels

Vous devez configurer les autorisations de manière à permettre à une entité IAM (utilisateur ou rôle) de baliser des appareils MFA virtuels. Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListMFADeviceTags`
- `iam:TagMFADevice`
- `iam:UntagMFADevice`

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter, afficher la liste ou supprimer une balise pour un appareil MFA virtuel

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit gérer les balises. Utilisez votre numéro de compte et *<MFATokenID>* remplacez-le par le nom du périphérique MFA virtuel dont les tags doivent être gérés. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
        "iam:TagMFADevice",
        "iam:UntagMFADevice"
    ],
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter une balise à un dispositif MFA virtuel spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit ajouter, mais non pas supprimer, les balises d'un appareil MFA spécifique.

> **Note**
>
> L'action `iam:TagMFADevice` nécessite d'inclure également l'action `iam:ListMFADeviceTags`.

Pour utiliser cette politique, remplacez *<MFATokenID>* par le nom du périphérique MFA virtuel dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
        "iam:TagMFADevice"
    ],
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

Vous pouvez également utiliser une politique AWS gérée, par exemple IAMFullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les appareils MFA virtuels (AWS CLI or AWS API)

Vous pouvez afficher la liste, attacher ou supprimer des balises pour un appareil MFA virtuel. Vous pouvez utiliser l' AWS API AWS CLI ou pour gérer les balises d'un appareil MFA virtuel.

Pour répertorier les balises actuellement associées à un périphérique MFA virtuel (AWS CLI or AWS API)

- AWS CLI : aws iam list-mfa-device-tags
- AWS API : ListMFADeviceTags

Pour associer des balises à un périphérique MFA virtuel (AWS CLI or AWS API)

- AWS CLI : aws iam tag-mfa-device

- AWS API : appareil TagMFA

Pour supprimer des balises d'un périphérique MFA virtuel (AWS CLI or AWS API)

- AWS CLI : aws iam untag-mfa-device
- AWS API : UntagMFADevice

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Marquer les instances du gestionnaire d'accès aux comptes

Vous devez configurer les autorisations pour permettre à une entité IAM (rôles ou utilisateurs) de baliser les instances du gestionnaire d'accès aux comptes. Vous pouvez spécifier l'une ou l'ensemble des actions de balise d'accès au compte suivantes dans une politique IAM :

- `account-access:TagResource`
- `account-access:ListTagsForResource`
- `account-access:UntagResource`

Pour autoriser une entité IAM (rôle ou utilisateur) à ajouter, répertorier ou supprimer un tag pour une instance de gestionnaire d'accès au compte

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit gérer les balises. Utilisez votre numéro de compte et les ID d'instance de votre IAM Identity Center et des instances de votre gestionnaire d'accès au compte pour remplacer les espaces réservés. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques IAM (console)".

```
{
    "Effect": "Allow",
    "Action": [
        "account-access:TagResource",
        "account-access:ListTagsForResource",
```

```
        "account-access:UntagResource"
    ],
    "Resource": "<account_access_manager_ARN>"
}
```

Pour autoriser une entité IAM (rôle ou utilisateur) à ajouter une balise à une instance spécifique du gestionnaire d'accès au compte

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit ajouter, mais pas supprimer, des balises pour une instance de gestionnaire d'accès au compte spécifique.

> **Note**
>
> L'action `account-access:TagResource` nécessite d'inclure également l'action `account-access:ListTagsForResource`.

Utilisez votre numéro de compte et les ID d'instance de votre IAM Identity Center et des instances de votre gestionnaire d'accès au compte pour remplacer les espaces réservés. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques IAM (console)".

```
{
    "Effect": "Allow",
    "Action": [
        "account-access:TagResource",
        "account-access:ListTagsForResource"
    ],
    "Resource": "<account_access_manager_ARN>"
}
```

Gestion des balises sur le gestionnaire d'accès au compte (console)

Vous pouvez gérer les balises d'une instance de gestionnaire d'accès au compte à partir du Console de gestion AWS.

Pour gérer les balises sur le gestionnaire d'accès au compte (console)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.

2. Dans le volet de navigation de la console, choisissez Gestionnaire d'accès au compte, puis choisissez l'onglet Paramètres.
3. Dans la section Tags, vous pouvez voir les tags actuellement configurés. Choisissez Gérer les balises, puis effectuez l'une des actions suivantes :
   - Choisissez Ajouter une nouvelle étiquette pour ajouter une nouvelle étiquette.
   - Modifiez les clés et les valeurs de balise existantes.
   - Choisissez Supprimer pour supprimer une étiquette.
4. Ajoutez ou supprimez des balises pour terminer l'ensemble de balises. Ensuite, choisissez Enregistrer les modifications.

Gestion des balises sur le gestionnaire d'accès au compte (AWS CLI or AWS API)

Vous pouvez répertorier, joindre ou supprimer des balises pour les instances du gestionnaire d'accès aux comptes à l'aide de l' AWS API AWS CLI or.

Pour répertorier les balises actuellement associées à une instance de gestionnaire d'accès aux comptes (AWS CLI ou AWS API)

- AWS CLI: `aws account-access list-tags-for-resource`
- AWS API : `account-access:ListTagsForResource`

Pour associer des balises à une instance de gestionnaire d'accès au compte (AWS CLI ou à une AWS API)

- AWS CLI: `aws account-access tag-resource`
- AWS API : `TagResource`

Pour supprimer des balises d'une instance de gestionnaire d'accès au compte (AWS CLI ou d'une AWS API)

- AWS CLI: `aws account-access untag-resource`
- AWS API : `UntagResource`

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez the section called "Variables et balises".

Transmettre les balises de session AWS STS

Les balises de session sont des attributs de paire clé-valeur que vous transmettez lorsque vous endossez un rôle IAM ou fédérez un utilisateur dans AWS STS. Pour ce faire, vous pouvez effectuer une AWS CLI demande d' AWS API via AWS STS ou via votre fournisseur d'identité (IdP). Lorsque vous demandez des informations AWS STS d'identification de sécurité temporaires, vous générez une session. Les sessions expirent et disposent d'informations d'identification, telles qu'une paire de clés d'accès et un jeton de session. Lorsque vous utilisez les informations d'identification de session pour effectuer une demande ultérieure, le contexte de demande inclut la clé de contexte `aws:PrincipalTag`. Vous pouvez utiliser la clé `aws:PrincipalTag` dans l'élément `Condition` de vos politiques pour autoriser ou refuser l'accès en fonction de ces balises.

Lorsque vous utilisez des informations d'identification temporaires pour effectuer une demande, votre principal peut inclure un ensemble d'étiquettes. Ces balises proviennent des sources suivantes :

1. Tags de session : balises transmises lorsque vous assumez le rôle ou que vous fédérez l'utilisateur à l'aide de l' AWS API AWS CLI ou. Pour plus d'informations sur ces opérations, consultez Opérations de balisage de session.
2. Incoming transitive session tags (Balises de session transitive entrantes) : ces balises ont été héritées d'une session précédente dans une chaîne de rôles. Pour plus d'informations, consultez Chaînage des rôles avec des balises de session plus loin dans cette rubrique.
3. Balises IAM : balises attachées à votre rôle IAM endossé.

Rubriques

- Opérations de balisage de session
- Choses à savoir sur les balises de session
- Autorisations requises pour ajouter des balises de session
- Transmettre des balises de session à l'aide AssumeRole
- Transmettre des balises de session à l'aide AssumeRoleWithSAML
- Transmettre des balises de session à l'aide AssumeRoleWithWebIdentity
- Transmettre des balises de session à l'aide GetFederationToken
- Chaînage des rôles avec des balises de session

- Utilisation des balises de session pour ABAC
- Affichage des balises de session dans CloudTrail

Opérations de balisage de session

Vous pouvez transmettre des balises de session à l'aide des opérations suivantes AWS CLI ou des opérations d' AWS API dans AWS STS. La fonctionnalité Console de gestion AWS Switch Role ne vous permet pas de transmettre des balises de session.

Vous pouvez également définir les balises de session comme transitives. Les balises transitives persistent pendant le chaînage des rôles. Pour de plus amples informations, veuillez consulter Chaînage des rôles avec des balises de session.

Le tableau suivant compare les méthodes de transmission des balises de session.

Opération	Qui peut endosser le rôle	Méthode pour transmettre des balises	Méthode pour définir des balises transitives
Opération d'interface de ligne de commande (CLI) `assume-role` ou d'API `AssumeRole`	Utilisateur ou session IAM	Paramètre API `Tags` ou option d'interface de ligne de commande (CLI) `--tags`	Paramètre API `TransitiveTagKeys` ou option d'interface de ligne de commande (CLI) `--transitive-tag-keys`
Opération d'interface de ligne de commande (CLI) `assume-role-with-saml` ou API `AssumeRoleWithSAML`	Tout utilisateur authentifié à l'aide d'un fournisseur d'identité SAML	Attribut SAML `PrincipalTag`	Attribut SAML `TransitiveTagKeys`
Opération d'interface de ligne de	Tout utilisateur authentifié à l'aide d'un fournisseur OIDC	Jeton `PrincipalTag` OIDC	Jeton `TransitiveTagKeys` OIDC

Opération	Qui peut endosser le rôle	Méthode pour transmettre des balises	Méthode pour définir des balises transitives
commande (CLI) `assume-role-with-web-identity` ou API `AssumeRoleWithWebIdentity`			
Opération d'interface de ligne de commande (CLI) `get-federation-token` ou API `GetFederationToken`	Utilisateur IAM ou utilisateur racine	Paramètre API `Tags` ou option d'interface de ligne de commande (CLI) `--tags`	Non pris en charge

Les opérations qui prennent en charge le balisage de session peuvent échouer si l'une des conditions suivantes est remplie :

- Vous transmettez plus de 50 balises de session.
- Le texte brut de vos clés de balise de session dépasse 128 caractères.
- Le texte brut des valeurs de vos balises de session dépasse 256 caractères.
- La taille totale du texte brut des politiques de session dépasse 2 048 caractères.
- Les politiques de session et les balises de session augmentent la taille du jeton de session. Si le jeton dépasse la limite de taille, l'opération échoue. Le message d'erreur indique la taille de votre jeton de session et la taille maximale autorisée, toutes deux en octets. Surveillez `SessionTokenUtilization` la réponse de AWS STS l'API pour suivre votre utilisation.

Choses à savoir sur les balises de session

Avant d'utiliser des balises de session, veuillez consulter les détails suivants concernant les sessions et les balises.

- Lorsque vous utilisez des balises de session, les politiques d'approbation pour tous les rôles connectés au fournisseur d'identité qui transmet des balises doivent disposer de l'autorisation `sts:TagSession`. Pour les rôles qui ne disposent pas de cette autorisation dans la politique d'approbation, l'opération `AssumeRole` échoue.
- Lorsque vous demandez une session, vous pouvez spécifier des balises de principal comme balises de session. Les balises s'appliquent aux demandes que vous effectuez à l'aide des informations d'identification de la session.
- Les balises utilisent des paires clé-valeur. Par exemple, pour ajouter des informations de contact à une session, vous pouvez ajouter la clé de balise de session `email` et la valeur de balise `johndoe@example.com`.
- Les balises de session doivent respecter les règles de dénomination des balises dans IAM et AWS STS. Cette rubrique contient des informations sur la sensibilité à la casse et les préfixes restreints qui s'appliquent à vos balises de session.
- Les nouvelles balises de session remplacent les balises de session de rôle ou d'utilisateur fédéré existantes avec la même clé de balise, quel que soit le cas de caractères.
- Vous ne pouvez pas transmettre de balises de session à l'aide du Console de gestion AWS.
- Les balises de session ne sont valides que pour la session en cours.
- Les balises de session prennent en charge le chaînage des rôles. Par défaut, AWS STS ne transmet pas de balises aux sessions de rôle suivantes. Toutefois, vous pouvez définir les balises de session comme transitives. Les balises transitives persistent durant le chaînage des rôles et remplacent la mise en correspondance de valeurs `ResourceTag` après l'évaluation de la politique de confiance de rôle. Pour de plus amples informations, veuillez consulter Chaînage des rôles avec des balises de session.
- Vous pouvez utiliser des balises de session pour contrôler l'accès aux ressources ou pour contrôler les balises qui peuvent être transmises à une session ultérieure. Pour plus d'informations, veuillez consulter Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC.
- Vous pouvez afficher les balises principales de votre session, y compris les balises de session, dans les journaux AWS CloudTrail . Pour de plus amples informations, veuillez consulter Affichage des balises de session dans CloudTrail.
- Vous devez transmettre une valeur unique pour chaque balise de session. AWS STS ne prend pas en charge les balises de session à valeurs multiples.
- Vous pouvez transmettre un maximum de 50 balises de session. Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

- Les politiques de session et les balises de session augmentent la taille du jeton de session. Si le jeton dépasse la limite de taille, l' AWS API AWS CLI or renvoie une erreur. Le message d'erreur indique la taille de votre jeton de session et la taille maximale autorisée, toutes deux en octets. L'élément de `SessionTokenUtilization` réponse indique le pourcentage de la limite de taille utilisé par le jeton. L'élément de `PackedPolicySize` réponse est obsolète. Utilisez `SessionTokenUtilization` à la place.

Autorisations requises pour ajouter des balises de session

En plus de l'action qui correspond à l'opération d'API, vous devez disposer de l'action d'autorisation suivante dans votre politique :

```
sts:TagSession
```

> **Important**
>
> Lorsque vous utilisez des balises de session, les politiques d'approbation de rôle pour tous les rôles connectés à un fournisseur d'identité doivent disposer de l'autorisation `sts:TagSession`. L'opération `AssumeRole` échoue pour tout rôle connecté à un fournisseur d'identité qui transmet des balises de session sans cette autorisation. Si vous ne souhaitez pas mettre à jour la politique d'approbation de rôle pour chaque rôle, vous pouvez utiliser une instance de fournisseur d'identité distincte pour transmettre les balises de session. Ajoutez ensuite l'autorisation `sts:TagSession` uniquement aux rôles qui sont connectés au fournisseur d'identité distinct.

Vous pouvez utiliser l'action `sts:TagSession` avec les clés de condition suivantes.

- `aws:PrincipalTag` : compare la balise attachée au principal effectuant la demande avec la balise spécifiée dans la politique. Par exemple, vous pouvez autoriser un principal à transmettre des balises de session uniquement si le principal qui effectue la demande possède les balises spécifiées.
- `aws:RequestTag` : compare la paire clé-valeur de balise qui a été transmise dans la demande avec la paire de balises que vous indiquez dans la politique. Par exemple, vous pouvez autoriser le principal à transmettre les balises de session spécifiées, mais uniquement avec les valeurs spécifiées.

- `aws:ResourceTag` : compare la paire clé-valeur de balise que vous avez spécifiée dans la politique avec la paire clé-valeur attachée à la ressource. Par exemple, vous pouvez autoriser le principal à transmettre des balises de session uniquement si le rôle qu'il endosse inclut les balises spécifiées.
- `aws:TagKeys` : compare les clés de balise d'une demande avec les clés que vous avez spécifiées dans la politique. Par exemple, vous pouvez autoriser le principal à transmettre uniquement les balises de session avec les clés de balise spécifiées. Cette clé de condition limite l'ensemble maximal de balises de session pouvant être passé.
- `sts:TransitiveTagKeys` : compare les clés des balises de session transitive de la demande à celles spécifiées dans la politique. Par exemple, vous pouvez écrire une politique pour permettre à un principal de définir uniquement des balises spécifiques comme transitives. Les balises transitives persistent pendant le chaînage des rôles. Pour plus d'informations, veuillez consulter Chaînage des rôles avec des balises de session.

Par exemple, la politique d'approbation de rôle suivante permet à l'utilisateur `test-session-tags` d'endosser le rôle auquel la politique est attachée. Lorsque cet utilisateur assume le rôle, il doit utiliser l' AWS API AWS CLI or pour transmettre les trois balises de session requises et l'ID externe requis. En outre, l'utilisateur peut choisir de définir les balises `Department` et `Project` comme transitives.

Example Politique d'approbation de rôle pour les balises de session

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIamUserAssumeRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-
tags"},
            "Condition": {
              "StringLike": {
                    "aws:RequestTag/Project": "*",
                    "aws:RequestTag/CostCenter": "*",
                    "aws:RequestTag/Department": "*"
```

```
                },
                "StringEquals": {"sts:ExternalId": "Example987"}
            }
        },
        {
            "Sid": "AllowPassSessionTagsAndTransitive",
            "Effect": "Allow",
            "Action": "sts:TagSession",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-
tags"},
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/Project": "*",
                    "aws:RequestTag/CostCenter": "*"
                },
                "StringEquals": {
                    "aws:RequestTag/Department": [
                        "Engineering",
                        "Marketing"
                    ]
                },
                "ForAllValues:StringEquals": {
                    "sts:TransitiveTagKeys": [
                        "Project",
                        "Department"
                    ]
                }
            }
        }
    ]
}
```

À quoi sert cette politique ?

- La instruction `AllowIamUserAssumeRole` permet à l'utilisateur `test-session-tags` d'endosser le rôle auquel la politique est attachée. Lorsque cet utilisateur endosse le rôle, il doit passer les balises de session et l'ID externe requis.
  - Le premier bloc de condition de cette instruction nécessite que l'utilisateur transmette les balises de session `Project`, `CostCenter` et `Department`. Les valeurs de balise n'ont pas d'importance dans cette instruction, vous pouvez donc utiliser des caractères génériques (*) pour

les valeurs de balise. Ce bloc garantit que l'utilisateur transmette au moins ces trois balises de session. Sinon, l'opération échoue. L'utilisateur peut transmettre des balises supplémentaires.

- Le deuxième bloc de condition nécessite que l'utilisateur transmette un ID externe avec la valeur `Example987`.

- L'instruction `AllowPassSessionTagsAndTransitive` autorise l'action d'autorisations `sts:TagSession`. Cette action doit être autorisée avant que l'utilisateur puisse transmettre des balises de session. Si votre politique inclut la première instruction sans la deuxième, l'utilisateur ne peut pas endosser le rôle.
  - Le premier bloc de condition de cette instruction permet à l'utilisateur de transmettre n'importe quelle valeur pour les balises de session `CostCenter` et `Project`. Pour ce faire, utilisez des caractères génériques (*) pour la valeur de la balise dans la politique, ce qui nécessite l'utilisation de l'opérateur de StringLike condition.
  - Le deuxième bloc de condition permet à l'utilisateur de transmettre uniquement la valeur `Engineering` ou `Marketing` de la balise de session `Department`.
  - Le troisième bloc de condition répertorie l'ensemble maximal de balises que vous pouvez définir comme transitives. L'utilisateur peut choisir de définir un sous-ensemble ou aucune balise comme transitive. Il ne peut pas définir de balises supplémentaires comme transitives. Vous pouvez exiger qu'il définisse au moins une des balises comme transitive en ajoutant un autre bloc de condition qui inclut `"Null":{"sts:TransitiveTagKeys":"false"}`.

Transmettre des balises de session à l'aide AssumeRole

L'`AssumeRole`opération renvoie un ensemble d'informations d'identification temporaires que vous pouvez utiliser pour accéder aux AWS ressources. Vous pouvez utiliser l'utilisateur ou les informations d'identification du rôle IAM pour appeler `AssumeRole`. Pour transmettre des balises de session tout en assumant un rôle, utilisez l'`--tags` AWS CLI option ou le paramètre `Tags` AWS API.

Pour définir les balises comme transitives, utilisez l'`--transitive-tag-keys` AWS CLI option ou le paramètre `TransitiveTagKeys` AWS API. Les balises transitives persistent pendant le chaînage des rôles. Pour plus d'informations, veuillez consulter Chaînage des rôles avec des balises de session.

L'exemple suivant montre un exemple de demande qui utilise `AssumeRole`. Dans cet exemple, lorsque vous endossez le rôle `my-role-example`, vous créez une session nommée `my-session`. Vous ajoutez les paires clé-valeur de balise de session `Project` = `Automation`, `CostCenter` = `12345` et `Department` = `Engineering`. Vous définissez également les balises `Project` et

`Department` comme transitives en spécifiant leurs clés. Vous devez transmettre une valeur unique pour chaque balise de session. AWS STS ne prend pas en charge les balises de session à valeurs multiples.

Example Exemple de requête AssumeRole CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/my-role-example \
--role-session-name my-session \
--tags Key=Project,Value=Automation Key=CostCenter,Value=12345
 Key=Department,Value=Engineering \
--transitive-tag-keys Project Department \
--external-id Example987
```

Transmettre des balises de session à l'aide AssumeRoleWithSAML

L'`AssumeRoleWithSAML`opération s'authentifie auprès de la SAML-based fédération. Cette opération renvoie un ensemble d'informations d'identification temporaires que vous pouvez utiliser pour accéder aux AWS ressources. Pour plus d'informations sur l'utilisation SAML-based de la fédération pour Console de gestion AWS l'accès, consultezPermettre aux principaux fédérés SAML 2.0 d'accéder au Console de gestion AWS. Pour plus de détails sur AWS l'accès à l'API AWS CLI ou à l'API, consultezFédération SAML 2.0. Pour un didacticiel sur la configuration de la fédération SAML pour vos utilisateurs Active Directory, consultez la section Authentification AWS fédérée avec Active Directory Federation Services (ADFS) dans le AWS blog de sécurité.

En tant qu'administrateur, vous pouvez autoriser les membres de l'annuaire de votre entreprise à se fédérer pour AWS utiliser l' AWS STS `AssumeRoleWithSAML`opération. Pour ce faire, effectuez les tâches suivantes :

1. Configurer votre réseau en tant que fournisseur SAML pour l'interface AWS
2. Créer un fournisseur SAML dans IAM
3. Création d'un rôle pour la fédération SAML 2.0 (console)
4. Achever la configuration de l'IdP SAML et créer des assertions pour la réponse d'authentification SAML

AWS inclut des fournisseurs d'identité dotés d'une expérience de bout en bout certifiée pour les balises de session avec leurs solutions d'identité. Pour savoir comment utiliser ces fournisseurs

d'identité pour configurer des balises de session, veuillez consulter Intégrez des fournisseurs de solutions SAML tiers à AWS.

Pour transmettre les attributs SAML en tant que balises de session, incluez l'élément `Attribute` dont l'attribut `Name` est défini sur `https://aws.amazon.com/SAML/Attributes/PrincipalTag:{TagKey}`. Utilisez l'élément `AttributeValue` pour spécifier la valeur de la balise. Incluez un élément `Attribute` distinct pour chaque balise de session.

Par exemple, supposons que vous souhaitez transmettre les attributs d'identité suivants en tant que balises de session :

- `Project:Automation`
- `CostCenter:12345`
- `Department:Engineering`

Pour transmettre ces attributs, incluez les éléments suivants dans votre assertion SAML.

Example Extrait d'une assertion SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Automation</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Department">
  <AttributeValue>Engineering</AttributeValue>
</Attribute>
```

Pour définir les balises précédentes ci-dessus comme transitives, incluez un autre élément `Attribute` dont l'attribut `Name` est défini sur `https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys`. Les balises transitives persistent pendant le chaînage des rôles. Pour plus d'informations, veuillez consulter Chaînage des rôles avec des balises de session.

Pour définir les balises `Department` et `Project` comme transitives, utilisez l'attribut à valeurs multiples suivant :

Example Extrait d'une assertion SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
```

```
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>Department</AttributeValue>
</Attribute>
```

Transmettre des balises de session à l'aide AssumeRoleWithWebIdentity

Utilisez la fédération conforme à OpenID Connect (OIDC) pour authentifier l'opération `AssumeRoleWithWebIdentity`. Cette opération renvoie un ensemble d'informations d'identification temporaires que vous pouvez utiliser pour accéder aux AWS ressources. Pour plus d'informations sur l'utilisation de la fédération d'identités Web pour Console de gestion AWS l'accès, consultezFédération OIDC.

Pour transmettre les balises de session depuis OpenID Connect (OIDC), vous devez inclure les balises de session dans le JWT (JSON Web Token) lorsque vous soumettez la requête `AssumeRoleWithWebIdentity`. Pour en savoir plus sur les jetons OIDC et les revendications, veuillez consulter Utilisation des jetons avec les groupes d'utilisateurs dans le Guide du développeur Amazon Cognito .

AWS prend en charge deux formats de réclamation pour inclure des balises de session dans le JWT :

- Format de demande imbriqué
- Format de demande aplati

Format de demande imbriqué

Le format de demande imbriqué utilise une structure au sein de l'espace de noms `https://aws.amazon.com/tags` du JWT. Dans ce format :

- Les balises des principaux sont représentées sous la forme d'un objet imbriqué sous la clé `principal_tags`.
- Chaque balise de principal est une valeur de chaîne unique.
- Les clés de balise transitives sont représentées dans un tableau sous la clé `transitive_tag_keys`.
- `principal_tags` et `transitive_tag_keys` sont imbriquées sous l'espace de noms `https://aws.amazon.com/tags`.

L'exemple suivant illustre un JWT décodé utilisant le format d'objet imbriqué :

Example Exemple de jeton Web JSON décodé utilisant le format de demande imbriqué

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/tags": {
        "principal_tags": {
            "Project": ["Automation"],
            "CostCenter": ["987654"],
            "Department": ["Engineering"]
        },
        "transitive_tag_keys": [
            "Project",
            "CostCenter"
        ]
    }
}
```

Format de demande aplati

Le format de demande aplati est compatible avec les fournisseurs d'identité qui ne prennent pas en charge les objets imbriqués dans les demandes JWT, tels que Microsoft Entra ID. Dans ce format :

- Les balises de principaux sont représentées sous forme de demandes distinctes avec le préfixe `https://aws.amazon.com/tags/principal_tags/`.
- Chaque balise de principal est une valeur de chaîne unique.
- Les clés de balise transitives sont représentées dans une seule demande sous la forme d'un tableau de chaînes avec le préfixe `https://aws.amazon.com/tags/transitive_tag_keys`.

Voyons maintenant comment les mêmes informations sont représentées à l'aide du format de demande aplati :

Example Exemple de jeton Web JSON décodé utilisant le format de demande aplati

```
{
```

```
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/tags/principal_tags/Project": "Automation",
    "https://aws.amazon.com/tags/principal_tags/CostCenter": "987654",
    "https://aws.amazon.com/tags/principal_tags/Department": "Engineering",
    "https://aws.amazon.com/tags/transitive_tag_keys": [
        "Project",
        "CostCenter"
    ]
}
```

Les deux exemples JWT décodés montrent un appel à `AssumeRoleWithWebIdentity` avec les balises de session `Project`, `CostCenter`, et `Department`. Les deux jetons définissent les balises `Project` et `CostCenter` comme transitives. Les balises transitives persistent pendant le chaînage des rôles. Pour de plus amples informations, veuillez consulter Chaînage des rôles avec des balises de session.

Le format de demande aplati permet d'obtenir le même résultat que le format de demande imbriqué, mais utilise une structure aplatie pour les balises. Il vous permet d'inclure des balises de session dans des environnements où les objets JSON imbriqués ne sont pas pris en charge dans les demandes JWT. Lorsque vous utilisez l'un ou l'autre format, assurez-vous que votre fournisseur d'identité est configuré pour émettre des jetons avec les structures de demande appropriées. AWS prend en charge les deux formats de demande, de sorte que vous pouvez choisir celui qui répond le mieux aux exigences spécifiques de votre fournisseur d'identité.

Transmettre des balises de session à l'aide GetFederationToken

Le `GetFederationToken` vous permet de fédérer votre utilisateur. Cette opération renvoie un ensemble d'informations d'identification temporaires que vous pouvez utiliser pour accéder aux AWS ressources. Pour ajouter des balises à votre session utilisateur fédérée, utilisez l'`--tags` AWS CLI option ou le paramètre `Tags` AWS API. Vous ne pouvez pas définir les balises de session comme transitives lorsque vous utilisez le `GetFederationToken`, car vous ne pouvez pas utiliser les informations d'identification temporaires pour endosser un rôle. Vous ne pouvez pas utiliser le chaînage de rôles dans ce cas.

Voici un exemple de réponse à l'aide de `GetFederationToken`. Dans cet exemple, lorsque vous demandez le jeton, vous créez une session nommée `my-fed-user`. Vous ajoutez les paires clé-valeur de la balise de session `Project` = `Automation` et `Department` = `Engineering`.

Example Exemple de requête GetFederationToken CLI

```
aws sts get-federation-token \
--name my-fed-user \
--tags key=Project,value=Automation key=Department,value=Engineering
```

Lorsque vous utilisez les informations d'identification temporaires renvoyées par l'opération `GetFederationToken`, les balises principales de la session incluent les balises de l'utilisateur et les balises de session transmises.

Chaînage des rôles avec des balises de session

Vous pouvez endosser un rôle, puis utiliser les informations d'identification temporaires pour en endosser un autre. Vous pouvez continuer d'une session à une autre. C'est ce qu'on appelle le chaînage des rôles. Lorsque vous transmettez des balises de session en endossant un rôle, vous pouvez définir les clés comme transitives. Cela garantit que ces balises de session sont transmises aux sessions suivantes dans une chaîne de rôles. Vous ne pouvez pas définir les balises de rôle comme transitives. Pour transmettre ces balises aux sessions suivantes, indiquez-les en tant que balises de session.

> **Note**
>
> Les balises transitives persistent durant le chaînage des rôles et remplacent la mise en correspondance de valeurs `ResourceTag` après l'évaluation de la politique de confiance de rôle.

L'exemple suivant montre comment AWS STS transmet les balises de session, les balises transitives et les balises de rôle aux sessions suivantes d'une chaîne de rôles.

Dans cet exemple de scénario de chaînage de rôles, vous utilisez une clé d'accès utilisateur IAM dans le AWS CLI pour assumer un rôle nommé. `Role1` Vous utilisez ensuite les informations d'identification de session résultantes pour endosser un second rôle nommé `Role2`. Vous pouvez ensuite utiliser les informations d'identification de la deuxième session pour endosser un troisième rôle nommé `Role3`. Ces demandes se déroulent sous la forme de trois opérations distinctes. Chaque

rôle est déjà balisé dans IAM. Et lors de chaque demande, vous transmettez des balises de session supplémentaires.
Enchaînement des rôles.

Lorsque vous enchaînez des rôles, vous pouvez vous assurer que les balises d'une session antérieure persistent jusqu'aux sessions ultérieures. Pour ce faire à l'aide de la commande de la CLI `assume-role`, vous devez transmettre la balise en tant que balise de session et la définir comme transitive. Vous transmettez la balise `Star` = 1 en tant que balise de session. La commande attache également la balise `Heart` = 1 au rôle et s'applique en tant que balise principale lorsque vous utilisez la session. Cependant, vous voulez également que la balise `Heart` = 1 transmette automatiquement à la deuxième ou troisième session. Pour ce faire, vous l'incluez manuellement en tant que balise de session. Les balises de principal de session résultantes incluent ces deux balises et les définissent comme transitives.
Assumer le premier rôle dans une chaîne de rôles.

Vous exécutez cette demande à l'aide de la AWS CLI commande suivante :

Example Exemple de requête AssumeRole CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role1 \
--role-session-name Session1 \
--tags Key=Star,Value=1 Key=Heart,Value=1 \
--transitive-tag-keys Star Heart
```

Vous utilisez ensuite les informations d'identification pour cette session pour endosser `Role2`. La commande attache la balise `Sun` = 2 au deuxième rôle et s'applique en tant que balise principale lorsque vous utilisez la deuxième session. Les `Star` balises `Heart` et héritent des balises de session transitives lors de la première session. Les balises principales résultantes de la deuxième session sont `Heart` = 1, `Star` = 1 et `Sun` = 2. `Heart` et `Star` continueront d'être transitives. La balise `Sun` attachée à `Role2` n'est pas marquée comme transitive, car il ne s'agit pas d'une balise de session. Les sessions futures n'héritent pas de cette balise.
Assumer le deuxième rôle dans une chaîne de rôles.

Vous effectuez cette deuxième requête à l'aide de la AWS CLI commande suivante :

Example Exemple de requête AssumeRole CLI

```
aws sts assume-role \
```

```
--role-arn arn:aws:iam::123456789012:role/Role2 \
--role-session-name Session2
```

Vous utilisez ensuite les informations d'identification de la deuxième session pour endosser `Role3`. Les balises principales de la troisième session proviennent de toutes les nouvelles balises de session, des balises de session transitive héritées et des balises de rôle. Les balises `Heart = 1` et `Star = 1` de la deuxième session ont été héritées de la balise de session transitive de la première session. Si vous essayez de transmettre la balise de session `Sun = 2`, l'opération échoue. La balise de session `Star = 1` héritée remplace la balise du rôle `Star = 3`. Dans le chaînage des rôles, la valeur d'une balise transitive remplace le rôle correspondant à la valeur `ResourceTag` avant l'évaluation de la politique de confiance de rôle. Dans cet exemple, si `Role3` utilise `Star` en tant que `ResourceTag` dans la politique de confiance de rôle, et définit la valeur `ResourceTag` à la valeur de balise transitive à partir de la session de rôle appelant. La `Lightning` balise de rôle s'applique également à la troisième session et n'est pas définie comme transitive.
Assumer le troisième rôle dans une chaîne de rôles.

Vous exécutez la troisième requête à l'aide de la AWS CLI commande suivante :

Example Exemple de requête AssumeRole CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role3 \
--role-session-name Session3
```

Utilisation des balises de session pour ABAC

Attribute-based le contrôle d'accès (ABAC) utilise une stratégie d'autorisation qui définit les autorisations en fonction des attributs des balises.

Si votre entreprise utilise un OIDC ou un fournisseur SAML-based d'identité (IdP) pour gérer les identités des utilisateurs, vous pouvez configurer votre assertion pour transmettre des balises de session à. AWS Par exemple, dans le cas des identités utilisateur d'entreprise, lorsque vos employés se fédérent AWS, leurs attributs sont AWS appliqués au principal qui en résulte. Vous pouvez ensuite utiliser l'ABAC pour autoriser ou refuser des autorisations sur la base de ces attributs. Pour en savoir plus, consultez Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC.

Pour plus d'informations sur l'utilisation d'IAM Identity Center avec ABAC, consultez la section Attributs pour le contrôle d'accès dans le Guide de l'utilisateur de AWS IAM Identity Center .

Affichage des balises de session dans CloudTrail

Vous pouvez l'utiliser AWS CloudTrail pour afficher les demandes utilisées pour assumer des rôles ou fédérer des utilisateurs. Le fichier journal CloudTrail contient des informations sur les balises principales de la session utilisateur endossée ou fédérée. Pour de plus amples informations, veuillez consulter Journalisation IAM et AWS STS Appels d'API avec AWS CloudTrail.

Supposons, par exemple, que vous fassiez une AWS STS `AssumeRoleWithSAML` demande, que vous transmettiez des balises de session et que vous définissiez ces balises comme transitives. Vous trouverez les informations suivantes dans votre journal CloudTrail .

Example Exemple de AssumeRoleWithSAML CloudTrail journal

```
    "requestParameters": {
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyRoleSessionName",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "durationSeconds": 3600,
        "roleArn": "arn:aws:iam::123456789012:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::123456789012:saml-provider/Shibboleth"
    },
```

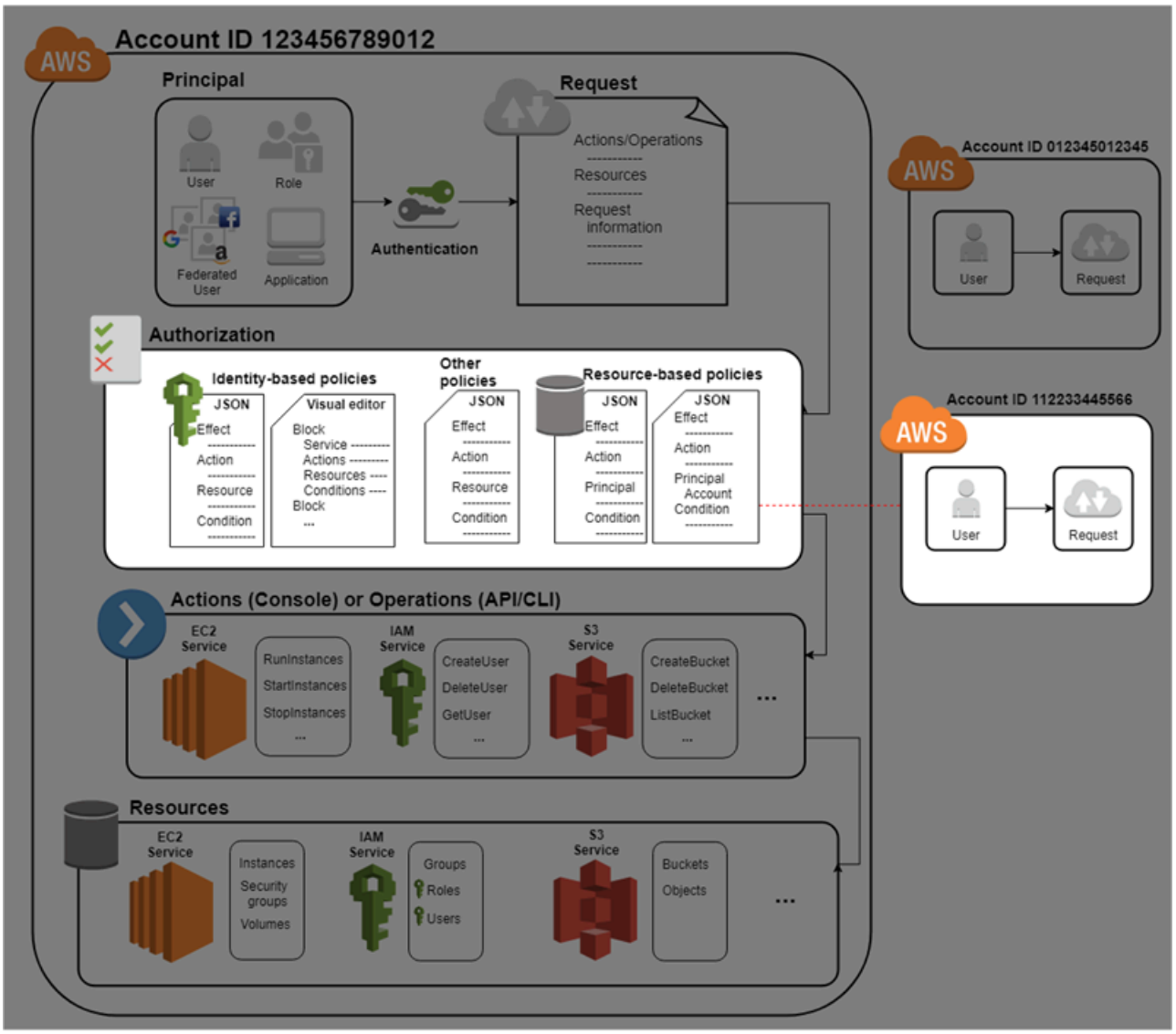
Vous pouvez consulter les exemples de CloudTrail journaux suivants pour visualiser les événements qui utilisent des balises de session.

- Exemple AWS STS événement d'API de chaînage de rôles dans le fichier CloudTrail journal
- Exemple SAML AWS STS Événement d'API dans le fichier CloudTrail journal
- Exemple OIDC AWS STS Événement d'API dans le fichier CloudTrail journal

Gestion des accès pour AWS resources

Gestion des identités et des accès AWS (IAM) est un service Web qui vous permet de contrôler en toute sécurité l'accès aux AWS ressources. Lorsqu'un mandant fait une demande d'accès AWS, le code d' AWS application vérifie si le mandant est authentifié (connecté) et autorisé (possède des autorisations). Vous gérez l'accès en AWS créant des politiques et en les associant à des identités ou à des AWS ressources IAM. Les politiques sont des documents JSON dans AWS la mesure où, lorsqu'ils sont attachés à une identité ou à une ressource, ils définissent leurs autorisations. Pour plus d'informations sur les types de politiques et les utilisations, consultez Politiques et autorisations dans Gestion des identités et des accès AWS.

Pour en savoir plus sur le reste du processus d'authentification et d'autorisation, consultez Fonctionnement de IAM.



Lors de l'autorisation, le code d' AWS application utilise les valeurs du contexte de la demande pour vérifier les politiques correspondantes et déterminer s'il faut autoriser ou refuser la demande.

AWS vérifie chaque politique qui s'applique au contexte de la demande. Si une seule politique refuse la demande, AWS refuse la totalité de la demande et arrête l'évaluation des politiques. Ceci est appelé un refus explicite. Étant donnée que les demandes sont refusées par défaut, IAM autorise votre demande uniquement si chacune de ses parties est autorisée par les politiques applicables. La logique d'évaluation pour une demande au sein d'un même compte utilise les règles suivantes :

- Par défaut, toutes les demandes sont implicitement refusées. (Autrement, par défaut, le Utilisateur racine d'un compte AWS dispose d'un accès complet.)
- Une autorisation explicite dans une politique basée sur l'identité ou les ressources remplace cette valeur par défaut.

- Si une limite d'autorisations, un AWS Organizations SCP ou une politique de session est présente, elle peut remplacer l'autorisation par un refus implicite.
- Un refus explicite dans n'importe quelle stratégie remplace toutes les autorisations.

Une fois que votre demande a été authentifiée et autorisée, elle est AWS approuvée. Si vous avez besoin d'effectuer une demande dans un compte différent, une politique dans l'autre compte doit vous permettre d'accéder à la ressource. En outre, l'entité IAM que vous utilisez pour effectuer la demande doit avoir une politique basée sur l'identité qui autorise la demande.

Vous pouvez également l'utiliser Délégation temporaire de l'IAM pour accorder à Amazon et à AWS ses partenaires un accès limité dans le temps pour configurer les AWS services de votre compte.

Ressources de gestion des accès

Pour plus d'informations sur les autorisations et la création de politiques, consultez les ressources suivantes :

Les articles suivants du blog sur la AWS sécurité décrivent les méthodes courantes pour rédiger des politiques d'accès aux compartiments et aux objets Amazon S3.

- Writing IAM Policies: How to Grant Access to an Amazon S3 Bucket
- Rédaction de politiques IAM : accorder l'accès aux User-Specific dossiers d'un compartiment Amazon S3
- IAM Policies and Bucket Policies and ACLs! Oh My! (Contrôler l'accès aux ressources S3)
- Présentation des Resource-Level autorisations RDS
- Démystifier les autorisations EC2 Resource-Level

Politiques et autorisations dans Gestion des identités et des accès AWS

Gérez l'accès en AWS créant des politiques et en les associant à des identités IAM (utilisateurs, groupes d'utilisateurs ou rôles) ou à des AWS ressources IAM. Une politique est un objet AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit ses autorisations. AWS évalue ces politiques lorsqu'un responsable IAM (utilisateur ou rôle) fait une demande. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée. La plupart des politiques sont

stockées AWS sous forme de documents JSON. AWS prend en charge neuf types de politiques : politiques basées sur l'identité, politiques basées sur les ressources, politiques sur les terminaux VPC, limites d'autorisations, politiques de contrôle des AWS Organizations services (SCP), politiques de contrôle AWS Organizations des ressources (RCP), listes de contrôle d'accès (ACL), partages Resource Access Manager (RAM) et politiques de session.

Les politiques IAM définissent les autorisations d'une action, quelle que soit la méthode que vous utilisez pour exécuter l'opération. Par exemple, si une politique autorise l'GetUseraction, un utilisateur utilisant cette politique peut obtenir des informations utilisateur à partir de l' Console de gestion AWS AWS CLI API ou de l' AWS API. Lorsque vous créez un utilisateur IAM, vous pouvez choisir d'autoriser la console ou un accès par programme. Si l'accès à la console est autorisé, l'utilisateur IAM peut s'y connecter à l'aide de ses informations d'identification. Si l'accès programmatique est autorisé, l'utilisateur peut utiliser des clés d'accès pour travailler avec la CLI ou l'API.

Types de politique

Les types de politiques suivants sont répertoriés dans l'ordre du plus fréquemment utilisé au moins fréquemment utilisé, et ils peuvent être utilisés dans AWS. Pour de plus amples informations, veuillez consulter les sections ci-dessous pour chaque type de politique.

- Identity-basedpolitiques : associez des politiques gérées et intégrées aux identités IAM (utilisateurs, groupes auxquels les utilisateurs appartiennent ou rôles). Identity-based les politiques accordent des autorisations à une identité.
- Resource-basedpolitiques : associez des politiques en ligne aux ressources. Les exemples les plus courants de politiques basées sur les ressources sont les politiques de compartiment Amazon S3 et les politiques de confiance des rôles IAM. Resource-based les politiques accordent des autorisations au principal spécifié dans la politique. Les principaux peuvent être dans le même compte que la ressource ou dans d'autres comptes.
- Politiques relatives aux terminaux VPC : attachez-les à un point de terminaison VPC pour contrôler quels principaux peuvent utiliser les politiques de point de terminaison VPC et quelles ressources sont accessibles via celui-ci. Les politiques relatives aux terminaux VPC constituent une limite d'accès supplémentaire limitée au trafic qui traverse le point de terminaison.
- Permissions boundaries (Limites d'autorisations) : utilisez une politique gérée en tant que limite d'autorisations pour une entité IAM (utilisateur ou rôle). Cette politique définit les autorisations maximales que les politiques basées sur une identité peuvent accorder à une entité, mais n'accorde pas d'autorisations. Les limites d'autorisations ne définissent pas les autorisations maximales qu'une politique basée sur les ressources peut accorder à une entité.

- AWS Organizations SCP : utilisez une politique de contrôle des AWS Organizations services (SCP) pour définir les autorisations maximales pour les utilisateurs IAM et les rôles IAM au sein des comptes de votre organisation ou de votre unité organisationnelle (OU). Les SCP limitent les autorisations que les politiques basées sur l'identité ou les politiques basées sur les ressources accordent aux utilisateurs IAM ou aux rôles IAM au sein du compte. Les SCP n'accordent pas d'autorisations.
- AWS Organizations RCP : utilisez une politique de contrôle AWS Organizations des ressources (RCP) pour définir les autorisations maximales pour les ressources au sein des comptes de votre organisation ou de votre unité organisationnelle (OU). Les RCP limitent les autorisations que les politiques basées sur l'identité ou sur les ressources peuvent accorder aux ressources des comptes au sein de votre organisation. Les RCP n'accordent pas d'autorisations.
- Access control lists (ACLs) (Listes de contrôle d'accès [ACL]) : utilisez les listes de contrôle d'accès (ACL) pour contrôler quels principaux dans d'autres comptes peuvent accéder à la ressource à laquelle la liste de contrôle d'accès (ACL) est attachée. Les listes de contrôle d'accès sont semblables aux politiques basées sur les ressources, bien qu'elles soient le seul type de politique qui n'utilise pas la structure d'un document de politique JSON. Les listes de contrôle d'accès sont des politiques d'autorisations entre comptes qui accordent des autorisations pour le principal spécifié. Les listes de contrôle d'accès ne peuvent pas accorder des autorisations aux entités au sein du même compte.
- AWS Partage de ressources RAM : utilisez AWS Resource Access Manager pour partager des ressources entre Comptes AWS des unités organisationnelles ou une organisation entière sans écrire de politiques individuelles basées sur les ressources pour chaque ressource partagée.
- Stratégies de session : transmettez des politiques de session avancées lorsque vous utilisez l' AWS API AWS CLI or pour assumer un rôle ou un utilisateur fédéré. Les politiques de session limitent les autorisations que les politiques basées sur l'identité du rôle ou de l'utilisateur accordent à la session. Les politiques de session limitent les autorisations d'une session créée, mais n'accordent pas d'autorisations. Pour plus d'informations, consultez Politiques de session.

Identity-based politiques

Identity-based les politiques sont des documents de politique d'autorisations JSON qui contrôlent les actions qu'une identité (utilisateurs, groupes d'utilisateurs et rôles) peut effectuer, sur quelles ressources et dans quelles conditions. Identity-based les politiques peuvent être classées plus en catégories :

- Politiques gérées : politiques autonomes basées sur l'identité que vous pouvez associer à plusieurs utilisateurs, groupes et rôles de votre. Compte AWS Il existe deux types de politiques gérées.
  - AWS politiques gérées : politiques gérées créées et gérées par AWS.
  - Politiques gérées par le client : politiques gérées que vous créez et gérez dans votre Compte AWS. Les politiques gérées par le client fournissent un contrôle plus précis de vos politiques que les politiques AWS gérées.
- Politiques en ligne : politiques que vous ajoutez directement à un utilisateur unique, un groupe d'utilisateurs, ou un rôle. Les politiques en ligne maintiennent une relation un-à-un stricte entre une politique et une identité. Elles sont supprimées lorsque vous supprimez l'identité.

Pour savoir comment choisir entre des politiques gérées et en ligne, veuillez consulter Choix entre politiques gérées et politiques en ligne.

Resource-based politiques

Resource-based les politiques sont des documents de stratégie JSON que vous joignez à une ressource telle qu'un compartiment Amazon S3. Ces politiques accordent au principal spécifié l'autorisation d'effectuer des actions spécifiques sur cette ressource et définissent dans quelles conditions cela s'applique. Resource-basedles politiques sont des politiques en ligne. Il ne s'agit pas de politiques gérées basées sur les ressources.

Pour permettre un accès comptes multiples , vous pouvez spécifier un compte entier ou des entités IAM dans un autre compte en tant que principal dans une politique basée sur les ressources. L'ajout d'un principal intercompte à une politique basée sur les ressources ne représente qu'une partie de l'instauration de la relation d'approbation. Lorsque le principal et la ressource sont séparés Comptes AWS, vous devez également utiliser une politique basée sur l'identité pour accorder au principal l'accès à la ressource. Toutefois, si une politique basée sur des ressources accorde l'accès à un principal dans le même compte, aucune autre politique basée sur l'identité n'est requise. Pour obtenir des instructions détaillées sur l'octroi d'un accès entre services, veuillez consulter Tutoriel IAM : accès délégué via AWS comptes utilisant des rôles IAM.

Le service IAM prend en charge un seul type de politique basée sur les ressources, nommé politique d'approbation de rôle, qui est attaché à un rôle IAM. Un rôle IAM est à la fois une identité et une ressource qui prend en charge les politiques basées sur les ressources. C'est pour cette raison que vous devez associer une politique d'approbation et une politique basée sur une identité à un rôle IAM. Les politiques d'approbation définissent quelles entités principaux (comptes, utilisateurs, rôles et

utilisateurs fédérés) peuvent endosser le rôle. Pour en savoir plus sur la façon dont les rôles IAM sont différents d'autres politiques basées sur les ressources, consultez Accès intercompte aux ressources dans IAM.

Pour connaître les autres services qui prennent en charge les politiques basées sur les ressources, voir AWS services qui fonctionnent avec IAM. Pour en savoir plus sur politiques basées sur les ressources, voir Identity-based politiques et politiques basées sur les ressources. Pour savoir si les principaux des comptes situés en dehors de votre zone de confiance (organisation ou compte de confiance) ont accès à vos rôles, consultez Qu'est-ce que l'Analyseur d'accès IAM ?.

Politiques de point de terminaison d'un VPC

Une politique de point de terminaison VPC est une politique basée sur les ressources que vous attachez à un point de terminaison VPC pour contrôler quels principaux peuvent utiliser le point de terminaison et quelles ressources sont accessibles via celui-ci. Les politiques relatives aux terminaux VPC ne remplacent ni ne remplacent les politiques basées sur l'identité ou les politiques basées sur les ressources associées au service de destination. Elles constituent une limite d'accès supplémentaire limitée au trafic qui traverse le point de terminaison.

Les politiques relatives aux terminaux VPC sont utilisées pour garantir que seules les identités fiables de votre AWS Organizations organisation peuvent accéder à des ressources fiables via vos points de terminaison VPC. Si vous ne joignez pas de politique de point de terminaison personnalisée, AWS associez une politique par défaut qui autorise un accès complet. Pour plus d'informations sur les politiques relatives aux terminaux, consultez la section Contrôler l'accès aux terminaux VPC à l'aide des politiques relatives aux terminaux dans le AWS PrivateLink Guide.

Limites d'autorisations IAM

Une limite d'autorisations est une fonctionnalité avancée dans laquelle vous définissez les autorisations maximales qu'une politique basée sur les identités peut accorder à une entité IAM. Lorsque vous définissez une limite d'autorisations pour une entité, l'entité peut effectuer uniquement les actions autorisées par ses deux ses politiques basées sur l'identité et ses limites d'autorisations.

Si vous spécifiez un rôle, une session ou un utilisateur dans l'élément principal d'une politique basée sur les ressources, une autorisation explicite dans la limite des autorisations n'est pas requise. Toutefois, si vous spécifiez un ARN de rôle dans l'élément principal d'une politique basée sur les ressources, une autorisation explicite dans la limite des autorisations est requise. Dans les deux cas, un refus explicite dans la limite des autorisations est effectif. Un refus explicite dans l'une de

ces politiques annule l'autorisation. Pour plus d'informations sur les limites d'autorisations, consultez Limites d'autorisations pour les entités IAM.

AWS Organizations politiques de contrôle des services (SCP)

Si vous activez toutes les fonctionnalités d'une organisation, vous pouvez appliquer les politiques de contrôle des services (SCP) à l'un ou à l'ensemble de vos comptes. Les SCP sont des politiques JSON qui spécifient les autorisations maximales pour les utilisateurs IAM et les rôles IAM au sein des comptes d'une organisation ou d'une unité d'organisation (UO). Le SCP limite les autorisations accordées aux directeurs dans les comptes des membres, y compris chacun d'entre eux. Utilisateur racine d'un compte AWS Un rejet explicite dans l'une de ces politiques annule toute autorisation définie dans d'autres politiques.

Pour plus d'informations sur les SCP AWS Organizations et les SCP, consultez la section Politiques de contrôle des services (SCP) dans le Guide de l'AWS Organizations utilisateur.

AWS Organizations politiques de contrôle des ressources (RCP)

Si vous activez toutes les fonctions d'une organisation, vous pouvez utiliser les politiques de contrôle des ressources (RCP) pour appliquer de manière centralisée les contrôles d'accès aux ressources de plusieurs Comptes AWS. Les RCP sont des politiques JSON que vous pouvez utiliser pour définir le nombre maximum d'autorisations disponibles pour les ressources de vos comptes sans mettre à jour les politiques IAM associées à chaque ressource que vous possédez. Le RCP limite les autorisations pour les ressources des comptes membres et peut avoir un impact sur les autorisations effectives pour les identités Utilisateur racine d'un compte AWS, y compris celles qui appartiennent à votre organisation ou non. Un refus explicite dans toute RCP applicable annule toute autorisation définie dans d'autres politiques susceptibles d'être associées à des identités ou à des ressources individuelles.

Pour plus d'informations sur les AWS Organizations RCP et une liste de ceux Services AWS qui prennent en charge les RCP, consultez les politiques de contrôle des ressources (RCP) dans le Guide de l'AWS Organizations utilisateur.

Listes de contrôle d'accès (ACL)

Les listes de contrôle d'accès (ACL) sont des politiques de service qui vous permettent de contrôler quels principaux d'un autre compte peuvent accéder à une ressource. Les ACL ne peuvent pas être utilisées pour contrôler l'accès pour un principal au sein du même compte. Les listes de contrôle d'accès sont semblables aux politiques basées sur les ressources, bien qu'elles soient le seul type de

politique qui n'utilise pas le format de document de politique JSON. Amazon S3 et Amazon VPC sont des exemples de services qui prennent en charge les ACL. AWS WAF Pour en savoir plus sur les listes de contrôle d'accès, consultez Présentation des listes de contrôle d'accès (ACL) dans le Guide du développeur Amazon Simple Storage Service.

AWS Gestionnaire d'accès aux ressources (AWS partages de ressources (RAM)

AWS Le Gestionnaire d'accès aux ressources (AWS RAM) vous permet de partager les ressources que vous créez dans l'une d'entre elles Compte AWS avec d'autres Comptes AWS, des unités organisationnelles ou l'ensemble d'une organisation AWS Organizations, sans écrire de politiques individuelles basées sur les ressources pour chaque ressource partagée. Bien que vous puissiez accorder un accès entre comptes en associant une politique basée sur les ressources directement à une ressource, AWS RAM fournit une alternative gérée et centralisée qui élimine le besoin d'énumérer les identifiants de compte dans les politiques ou de conserver des documents de politique identiques pour tous les comptes.

Grâce à la RAM, les comptes utilisateurs voient les ressources partagées de manière native dans leurs consoles de service, les propriétaires des ressources conservent la pleine propriété et une visibilité sur les personnes qui y ont accès, et les autorisations gérées définissent le maximum d'actions que les consommateurs peuvent effectuer, le tout étant régi par un partage de ressources unique plutôt que par des politiques par ressource. Les cas d'utilisation courants incluent le partage de sous-réseaux VPC, de pièces jointes AWS Transit Gateway, de règles de résolution Route 53 et de configurations de License Manager entre les comptes. Pour plus d'informations, voir Qu'est-ce que AWS Resource Access Manager ? dans le guide de l'utilisateur de la AWS RAM.

Politiques de session

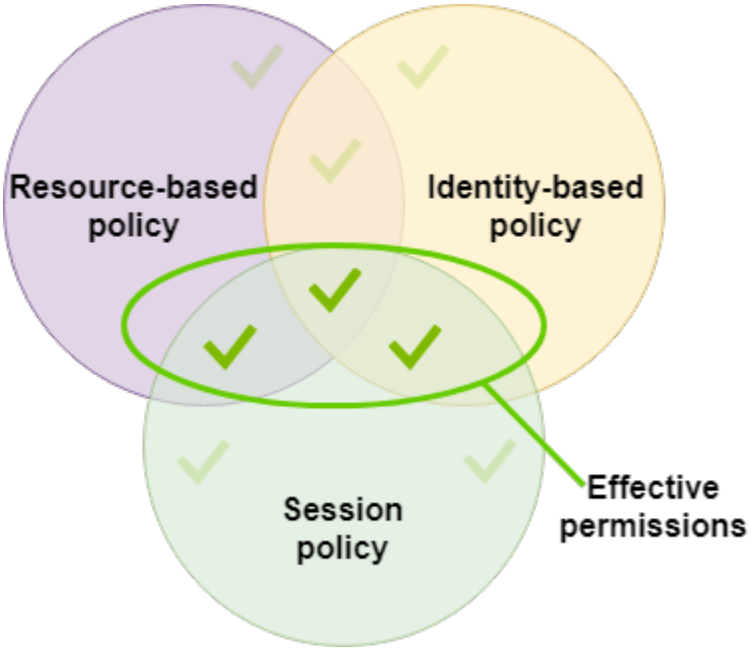
Les politiques de session sont des politiques avancées que vous transmettez en tant que paramètre lorsque vous créez par programmation une session temporaire pour un rôle ou un utilisateur principal AWS STS fédéré. Les autorisations d'une session sont une combinaison des politiques basées sur l'identité de l'entité IAM (utilisateur ou rôle) utilisée pour créer la session et les politiques de session. Les autorisations peuvent également provenir d'une politique basée sur les ressources. Un refus explicite dans l'une de ces politiques annule l'autorisation.

Vous pouvez créer une session de rôle et transmettre des politiques de session par programmation à l'aide des opérations d'API `AssumeRole`, `AssumeRoleWithSAML` ou `AssumeRoleWithWebIdentity`. Vous pouvez transmettre un seul document de politique de session en ligne JSON à l'aide du paramètre `Policy`. Vous pouvez utiliser le paramètre

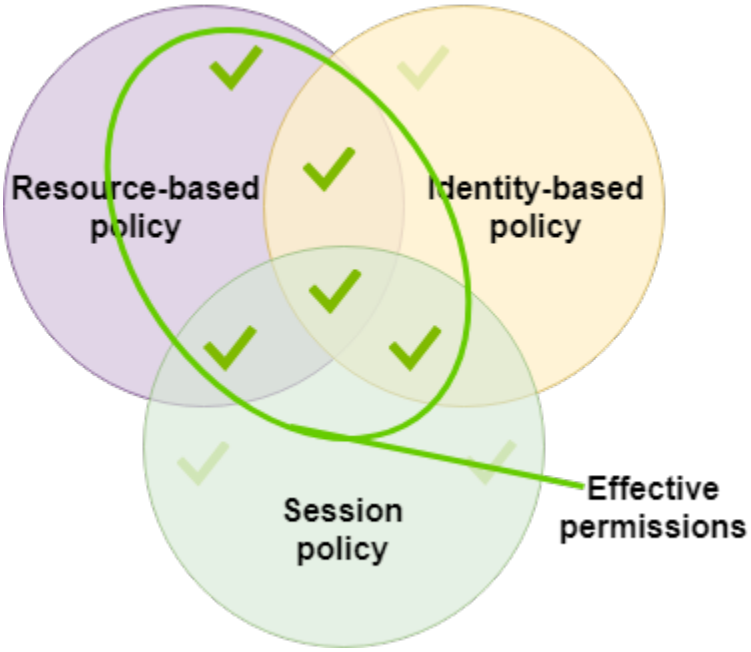
`PolicyArns` pour spécifier jusqu'à 10 politiques de session gérées. Pour plus d'informations sur la création d'une session de rôle, consultez Autorisations affectées aux informations d'identification de sécurité temporaires.

Lorsque vous créez une session principale d'utilisateur AWS STS fédéré, vous utilisez les clés d'accès de l'utilisateur IAM pour appeler par programmation l'opération d'API. `GetFederationToken` Vous devez également transmettre des politiques de session. Les autorisations de session obtenues sont une combinaison de la politique basée sur l'identité et de la politique de session. Pour plus d'informations sur la création d'une session d'utilisateur fédéré, consultez Demande d'informations d'identification via un fournisseur d'identité personnalisé.

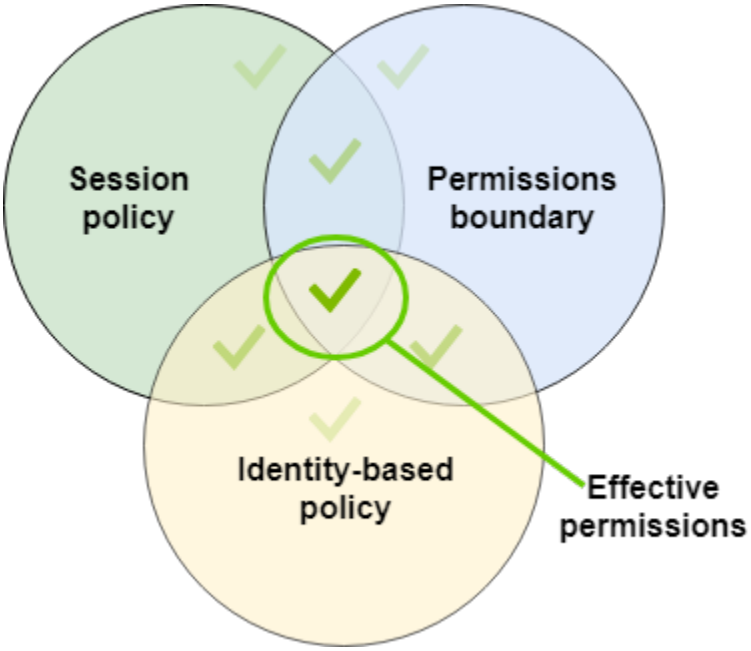
Une politique basée sur les ressources peut spécifier l'ARN de l'utilisateur ou du rôle en tant que principal. Dans ce cas, les autorisations de la politique basée sur les ressources sont ajoutées à la politique basée sur l'identité du rôle ou l'utilisateur avant la création de la session. La politique de session limite les autorisations totales accordées par la politique basée sur les ressources et la politique basée sur l'identité. Les autorisations de session résultantes sont l'intersection des politiques de session et des politiques basées sur les ressources, plus l'intersection des politiques de session et des politiques basées sur l'identité.



Une politique basée sur les ressources peut spécifier l'ARN de la session en tant que principal. Dans ce cas, les autorisations de la politique basée sur les ressources sont ajoutées après la création de la session. Les autorisations de politique basée sur les ressources ne sont pas limitées par la politique de session. La session résultante dispose de toutes les autorisations de la politique basée sur les ressources en plus de la combinaison de la politique basée sur l'identité et de la politique de session.



Une limite d'autorisations peut définir la limite maximale des autorisations pour un utilisateur ou un rôle qui est utilisé pour créer une session. Dans ce cas, les autorisations de session obtenues sont une combinaison de la politique de session, de la limite d'autorisations et de la politique basée sur l'identité. Toutefois, une limite d'autorisations ne limite pas les autorisations accordées par une politique basée sur les ressources, qui spécifie l'ARN de la session résultante.



Politiques et utilisateur racine

Elle Utilisateur racine d'un compte AWS est affectée par certains types de politiques mais pas par d'autres. Vous ne pouvez pas attacher des politiques basées sur l'identité à l'utilisateur racine, et vous ne pouvez pas définir la limite d'autorisations pour l'utilisateur racine. Pourtant, vous pouvez spécifier l'utilisateur racine en tant que principal dans une politique basée sur les ressources ou une liste de contrôle d'accès. Un utilisateur racine est toujours le membre d'un compte. Si ce compte est membre d'une organisation en AWS Organizations, l'utilisateur root est affecté par les SCP et les RCP du compte.

Présentation des politiques JSON

La plupart des politiques sont stockées AWS sous forme de documents JSON. Identity-based les politiques et les politiques utilisées pour définir les limites des autorisations sont des documents de stratégie JSON que vous attachez à un utilisateur ou à un rôle. Resource-based les politiques sont des documents de stratégie JSON que vous joignez à une ressource. Les SCP et les RCP sont des documents de politique JSON dotés d'une syntaxe restreinte que vous associez à la AWS Organizations« racine de l'organisation », à l'unité organisationnelle (OU) ou à un compte. Les listes de contrôle d'accès sont également attachées à une ressource, mais vous devez utiliser une syntaxe différente. Les politiques de session sont des politiques JSON que vous fournissez lorsque vous endossez une session de rôle ou d'utilisateur fédéré.

Il n'est pas nécessaire pour vous de comprendre la syntaxe JSON. Vous pouvez utiliser l'éditeur visuel du Console de gestion AWS pour créer et modifier des politiques gérées par les clients sans jamais utiliser JSON. Toutefois, si vous utilisez des politiques en ligne pour des groupes ou des politiques complexes, vous devez quand même créer et modifier ces politiques dans l'éditeur JSON à l'aide de la console. Pour plus d'informations sur l'utilisation de l'éditeur visuel, consultez Définition d’autorisations IAM personnalisées avec des politiques gérées par le client et Modification de politiques IAM.

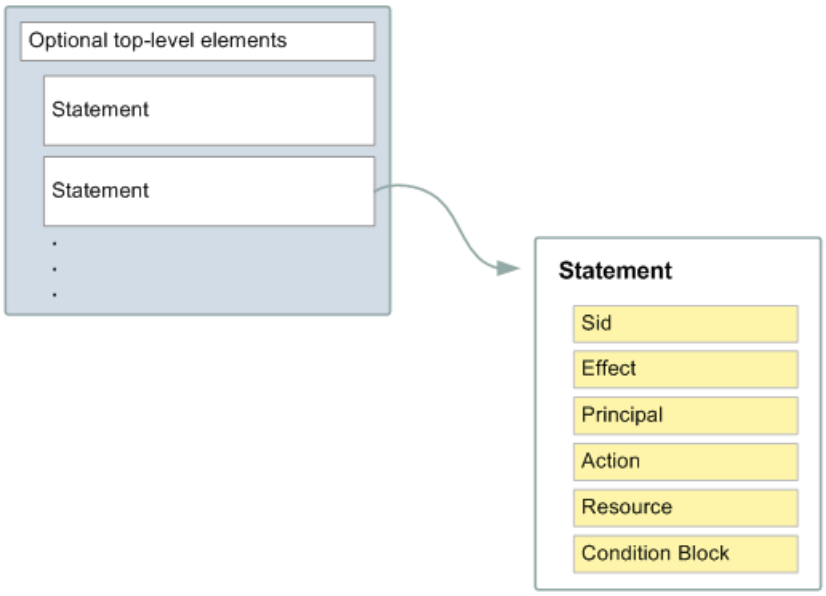
Lorsque vous créez ou modifiez une politique JSON, IAM peut effectuer une validation de politique pour vous aider à créer une politique efficace. IAM identifie les erreurs de syntaxe JSON, tandis que IAM Access Analyzer fournit des vérifications de politique supplémentaires avec des recommandations pour vous aider à affiner vos politiques. Pour en savoir plus sur la validation de politiques, veuillez consulter Validation de politique IAM. Pour en savoir plus sur les vérifications des politiques IAM Access Analyzer et les recommandations exploitables, veuillez consulter Validation de politique IAM Access Analyzer.

Structure d'un document de politique JSON

Comme illustré dans la figure suivante, un document de politique JSON inclut les éléments suivants :

- Informations facultatives sur l'ensemble de la politique (en haut du document)
- Une ou plusieurs instructions individuelles

Chaque instruction inclut des informations sur une seule autorisation. Si une politique comprend plusieurs déclarations, AWS applique une logique `OR` à toutes les déclarations lors de leur évaluation. Si plusieurs politiques s'appliquent à une demande, AWS applique une logique `OR` à toutes ces politiques lors de leur évaluation.



Les informations contenues dans une instruction sont situées dans une série d'éléments.

- Version : spécifiez la version du langage de politique que vous souhaitez utiliser. Nous vous recommandons de choisir la dernière version `2012-10-17` . Pour de plus amples informations, consultez Éléments de politique JSON IAM : Version.
- Statement : utilisez cet élément de politique principal comme conteneur pour les éléments suivants. Vous pouvez inclure plus d'une instruction dans une politique.
- Sid (Facultatif) : incluez un ID d'instruction facultatif pour différencier vos instructions.
- Effect : utilisez `Allow` ou `Deny` pour indiquer si la politique autorise ou refuse l'accès.
- Principal(Obligatoire dans certaines circonstances) — Si vous créez une politique basée sur les ressources, vous devez indiquer le compte, l'utilisateur, le rôle ou le principal utilisateur AWS

STS fédéré auquel vous souhaitez autoriser ou refuser l'accès. Si vous créez une politique d'autorisations IAM à attacher à un utilisateur ou un rôle, vous ne pouvez pas inclure cet élément. Le principal est implicitement cet utilisateur ou rôle.

- Action : incluez la liste des actions autorisées ou refusées par la politique.
- Resource (Obligatoire dans certaines circonstances) : si vous créez une politique d'autorisations IAM, vous devez spécifier la liste des ressources auxquelles les actions s'appliquent. Si vous créez une politique basée sur les ressources, cet élément est requis ou non en fonction de la ressource que vous utilisez.
- Condition (Facultatif) : spécifiez les circonstances dans lesquelles la politique accorde une autorisation.

Pour en savoir plus sur ces éléments et d'autres éléments de politique plus avancés, consultez Référence de l'élément de politique JSON IAM.

Plusieurs instructions et plusieurs politiques

Si vous souhaitez définir plusieurs autorisations pour une entité (utilisateur ou rôle), vous pouvez utiliser plusieurs instructions dans une seule politique. Vous pouvez également attacher plusieurs politiques. Si vous essayez de définir plusieurs autorisations dans une seule instruction, votre politique peut ne pas accorder l'accès que vous attendez. Nous vous recommandons de répartir les politiques par type de ressource.

En raison de la taille limitée des politiques, il peut être nécessaire d'en utiliser plusieurs pour les autorisations les plus complexes. Il est également conseillé de créer des regroupements fonctionnels d'autorisations dans une politique gérée par le client séparée. Par exemple, créez une politique pour la gestion des utilisateurs IAM, une pour la gestion automatique et une autre pour la gestion de compartiment S3. Quelle que soit la combinaison de plusieurs relevés et de plusieurs politiques, AWS évalue vos politiques de la même manière.

Par exemple, la politique suivante comporte trois instructions, chacune définissant un ensemble séparé d'autorisations dans un seul compte. Les instructions définissent les opérations suivantes :

- La première instruction, avec un `Sid` (ID d'instruction) de `FirstStatement`, permet à l'utilisateur disposant de la politique attachée de modifier son propre mot de passe. L'élément `Resource` de cette instruction a pour valeur « * » (ce qui signifie « toutes les ressources »). Toutefois, dans la pratique, l'opération d'API `ChangePassword` (ou la commande CLI `change-password` équivalente) affecte uniquement le mot de passe de l'utilisateur qui fait la demande.

- La deuxième instruction permet à l'utilisateur de répertorier tous les compartiments Amazon S3 de son Compte AWS. L'élément `Resource` de cette instruction a pour valeur `"*"` (ce qui signifie « toutes les ressources »). Cependant, du fait que les politiques n'accordent pas l'accès aux ressources des autres comptes, l'utilisateur ne peut répertorier que les compartiments de son propre Compte AWS.
- La troisième instruction permet à l'utilisateur de répertorier les objets situés dans un compartiment appelé `amzn-s3-demo-bucket-confidential-data`, mais uniquement lorsque l'utilisateur est authentifié avec une authentification multi-facteur (MFA). L'élément `Condition` de la politique applique l'authentification MFA.

  Lorsqu'une instruction de politique contient un élément `Condition`, l'instruction est effective uniquement lorsque l'élément `Condition` est true. Dans ce cas, la `Condition` valeur est évaluée à true lorsque l'utilisateur l'est MFA-authenticated. Si l'utilisateur ne l'est pas MFA-authenticated, `Condition` le résultat est faux. Dans ce cas, la troisième instruction de cette politique ne s'applique pas et l'utilisateur n'a pas accès au compartiment `amzn-s3-demo-bucket-confidential-data`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "FirstStatement",
      "Effect": "Allow",
      "Action": ["iam:ChangePassword"],
      "Resource": "*"
    },
    {
      "Sid": "SecondStatement",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
    {
      "Sid": "ThirdStatement",
      "Effect": "Allow",
      "Action": [
        "s3:List*",
```

```
        "s3:Get*"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket-confidential-data",
        "arn:aws:s3:::amzn-s3-demo-bucket-confidential-data/*"
      ],
      "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
    }
  ]
}
```

Exemples de syntaxe d'une politique JSON

La politique basée sur l'identité suivante permet au principal implicite de répertorier un seul compartiment Amazon S3 nommé `amzn-s3-demo-bucket` :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket"
  }
}
```

La politique basée sur les ressources suivante peut être attachée à un compartiment Amazon S3. La politique permet aux membres d'une entité spécifique Compte AWS d'effectuer toutes les actions Amazon S3 dans le compartiment nommé`amzn-s3-demo-bucket`. Elle permet à n'importe quelle action d'être exécutée sur un compartiment ou les objets qu'il contient. (Du fait que la politique accorde une approbation uniquement au compte, les utilisateurs individuels du compte doivent se voir accorder des autorisations concernant les actions Amazon S3 spécifiées.)

JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "1",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:root"
                ]
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

Pour afficher des exemples de politiques sur des scénarios courants, consultez Exemples de politiques basées sur l'identité IAM.

Accorder le moindre privilège

Lorsque vous créez des politiques IAM, suivez le conseil de sécurité standard du moindre privilège, principe selon lequel il ne faut accorder que les autorisations requises pour une seule tâche. Déterminez les actions que les utilisateurs (et les rôles) doivent effectuer et élaborez des politiques leur permettant de réaliser uniquement ces tâches.

Commencez avec un ensemble d'autorisations minimum et accordez-en d'autres si nécessaire. Il est plus sûr de procéder ainsi que de commencer par des autorisations trop clémentes pour ensuite essayer de les renforcer ultérieurement.

Comme alternative au moindre privilège, vous pouvez utiliser des politiques gérées par AWS ou des politiques accompagnées d'autorisations de * par caractère générique, pour commencer avec les politiques. Veuillez prendre en compte le risque de sécurité constitué par l'octroi, à vos principaux, de davantage d'autorisations que nécessaire pour accomplir leur tâche. Contrôlez ces principaux afin de savoir quelles autorisations ils utilisent. Écrivez ensuite des politiques de moindre privilège.

IAM propose plusieurs options pour vous aider à affiner les autorisations que vous octroyez.

- Understand access level groupings (Comprendre les regroupements de niveau d'accès) : vous pouvez utiliser des regroupements de niveaux d'accès pour comprendre le niveau d'accès accordé par la politique. Les actions de politique sont classées en tant que `List`, `Read`, `Write`, `Permissions management` ou `Tagging`. Par exemple, vous pouvez choisir des actions dans les niveaux d'accès `List` et `Read` pour accorder un accès en lecture seule à vos utilisateurs. Pour savoir comment utiliser les récapitulatifs de politiques pour comprendre les autorisations de niveau d'accès, consultez Niveaux d'accès dans les récapitulatifs de politique.
- Valider vos politiques : vous pouvez effectuer une validation de politique à l'aide d'IAM Access Analyzer lorsque vous créez et modifiez des politiques JSON. Nous vous recommandons d'examiner et de valider toutes vos politiques existantes. IAM Access Analyzer fournit plus de 100 vérifications de politiques pour valider vos politiques. Il génère des avertissements de sécurité lorsqu'une instruction de votre politique autorise un accès que nous considérons comme trop permissif. Vous pouvez utiliser les recommandations exploitables contenues dans les avertissements de sécurité lorsque vous travaillez à l'octroi du moindre privilège. Pour en savoir plus sur les vérifications de politique fournies par IAM Access Analyzer, veuillez consulter IAM Access Analyzer policy validation (Validation de politique IAM Access Analyzer).
- Générer une politique basée sur l'activité d'accès : pour affiner les autorisations que vous octroyez, vous pouvez générer une politique IAM basée sur l'activité d'accès pour une entité IAM (utilisateur ou rôle). IAM Access Analyzer examine vos AWS CloudTrail journaux et génère un modèle de politique contenant les autorisations qui ont été utilisées par l'entité au cours de la période que vous avez spécifiée. Vous pouvez utiliser le modèle pour créer une politique gérée avec des autorisations affinées, puis l'attacher à l'entité IAM. De cette façon, vous n'accordez que les autorisations dont l'utilisateur ou le rôle a besoin pour interagir avec les AWS ressources correspondant à votre cas d'utilisation spécifique. Pour en savoir plus, consultez Génération d'une politique de l'analyseur d'accès IAM.
- Utiliser les dernières informations consultées : les dernières informations consultées sont une autre fonction pouvant vous aider avec le principe de moindre privilège. Consultez ces informations sous l'onglet Access Advisor sur la page des détails de la console IAM pour un utilisateur, un groupe, un rôle ou une politique IAM. Les dernières informations consultées incluent également des informations sur les dernières actions consultées pour certains services, tels qu'Amazon EC2, IAM, Lambda et Amazon S3. Si vous vous connectez à l'aide AWS Organizations des informations d'identification du compte de gestion, vous pouvez consulter les informations relatives au dernier accès au service dans la AWS Organizations section de la console IAM. Vous pouvez également utiliser l' AWS API AWS CLI or pour récupérer un rapport contenant les dernières informations consultées pour les entités ou les politiques dans IAM ou AWS Organizations. Vous pouvez utiliser ces informations pour identifier les autorisations inutiles afin d'affiner votre IAM ou vos

AWS Organizations politiques afin de mieux respecter le principe du moindre privilège. Pour de plus amples informations, veuillez consulter Affiner les autorisations en AWS utilisant les dernières informations consultées.

- Consultez les événements de votre compte dans AWS CloudTrail — Pour réduire davantage les autorisations, vous pouvez consulter les événements de votre compte dans l'historique des AWS CloudTrail événements. CloudTrail les journaux d'événements contiennent des informations détaillées sur les événements que vous pouvez utiliser pour réduire les autorisations de la politique. Les journaux incluent uniquement les actions et les ressources dont vos entités IAM ont besoin. Pour plus d'informations, consultez la section Affichage CloudTrail des événements dans la CloudTrail console dans le Guide de AWS CloudTrail l'utilisateur.

Pour en savoir plus, consultez les rubriques de politiques pour des services individuels qui fournissent des exemples de rédaction de politiques pour des ressources spécifiques à des services.

- Utilisation de politiques basées sur les ressources pour DynamoDB dans le Guide du développeur Amazon DynamoDB
- Politiques de compartiment pour Amazon S3 dans le Guide de l'utilisateur Amazon Simple Storage Service.
- Présentation de la liste de contrôle d'accès (ACL) dans le Guide de l'utilisateur Amazon Simple Storage Service

Politiques gérées et politiques en ligne

Lorsque vous définissez les autorisations pour une identité dans IAM, vous devez décider si vous souhaitez utiliser une politique gérée par AWS , une politique gérée par le client ou une politique en ligne. Les rubriques suivantes contiennent d'autres informations sur chaque type de politique basée sur l'identité et sur la façon de les utiliser.

Le tableau suivant décrit ces politiques :

Type de politique	Description	Qui gère la politique ?	Modifier les autorisations ?	Nombre de principaux appliqués à la politique ?
AWS stratégies gérées	Politique autonome créée et administrée par AWS.	AWS	Non	Nombreux
Politiques gérées par le client	Vous créez des politiques pour des cas d'utilisation spécifiques et vous pouvez les modifier ou les mettre à jour aussi souvent que vous le souhaitez.	Vous	Oui	Nombreux
Politiques en ligne	Politique créée pour une identité IAM unique (utilisateur, groupe ou rôle) qui maintient une relation strictement individuelle entre une politique et une identité.	Vous	Oui	Un

Pour obtenir des instructions sur la création de rôles et l'association de ces politiques, consultezCréation des rôles et association des politiques (console).

Rubriques

- AWS stratégies gérées
- Politiques gérées par le client
- Politiques en ligne
- Choix entre politiques gérées et politiques en ligne
- Conversion d'une politique en ligne en politique gérée
- Obsolète AWS stratégies gérées

AWS stratégies gérées

Une politique gérée par AWS est une politique autonome qui est créée et gérée par AWS. Une politique autonome correspond à une politique qui possède son propre Amazon Resource Name (ARN) comprenant le nom de la politique. Par exemple, il `arn:aws:iam::aws:policy/IAMReadOnlyAccess` s'agit d'une politique AWS gérée. Pour plus d'informations sur les ARN, consultez ARN IAM. Pour obtenir la liste des politiques AWS gérées pour Services AWS, consultez la section Politiques AWS gérées.

AWS les politiques gérées vous permettent d'attribuer facilement les autorisations appropriées aux utilisateurs, aux groupes IAM et aux rôles. Plus rapide que d'écrire vous-même les politiques, cela inclut des autorisations pour de nombreux cas d'utilisation courants.

Vous ne pouvez pas modifier les autorisations définies dans les politiques AWS gérées. AWS met occasionnellement à jour les autorisations définies dans une politique AWS gérée. Dans ce cas AWS , la mise à jour affecte toutes les entités principales (utilisateurs IAM, groupes IAM et rôles IAM) auxquelles la politique est attachée. AWS est le plus susceptible de mettre à jour une politique AWS gérée lorsqu'un nouveau AWS service est lancé ou que de nouveaux appels d'API sont disponibles pour des services existants. Par exemple, la politique AWS gérée appelée ReadOnlyAccess fournit un accès en lecture seule à l'ensemble Services AWS des ressources. Lors du AWS lancement d'un nouveau service, AWS met à jour la ReadOnlyAccess politique pour ajouter des autorisations en lecture seule pour le nouveau service. Les autorisations mises à jour s'appliquent à toutes les entités du principal auxquelles la politique est attachée.

Politiques AWS gérées d'accès complet : elles définissent les autorisations pour les administrateurs de services en accordant un accès complet à un service. En voici quelques exemples :

- AmazonDynamoDBFullAccess

- IAMFullAccess

Power-user AWS politiques gérées : elles fournissent un accès complet aux Services AWS ressources, mais ne permettent pas de gérer les utilisateurs et les groupes IAM. En voici quelques exemples :

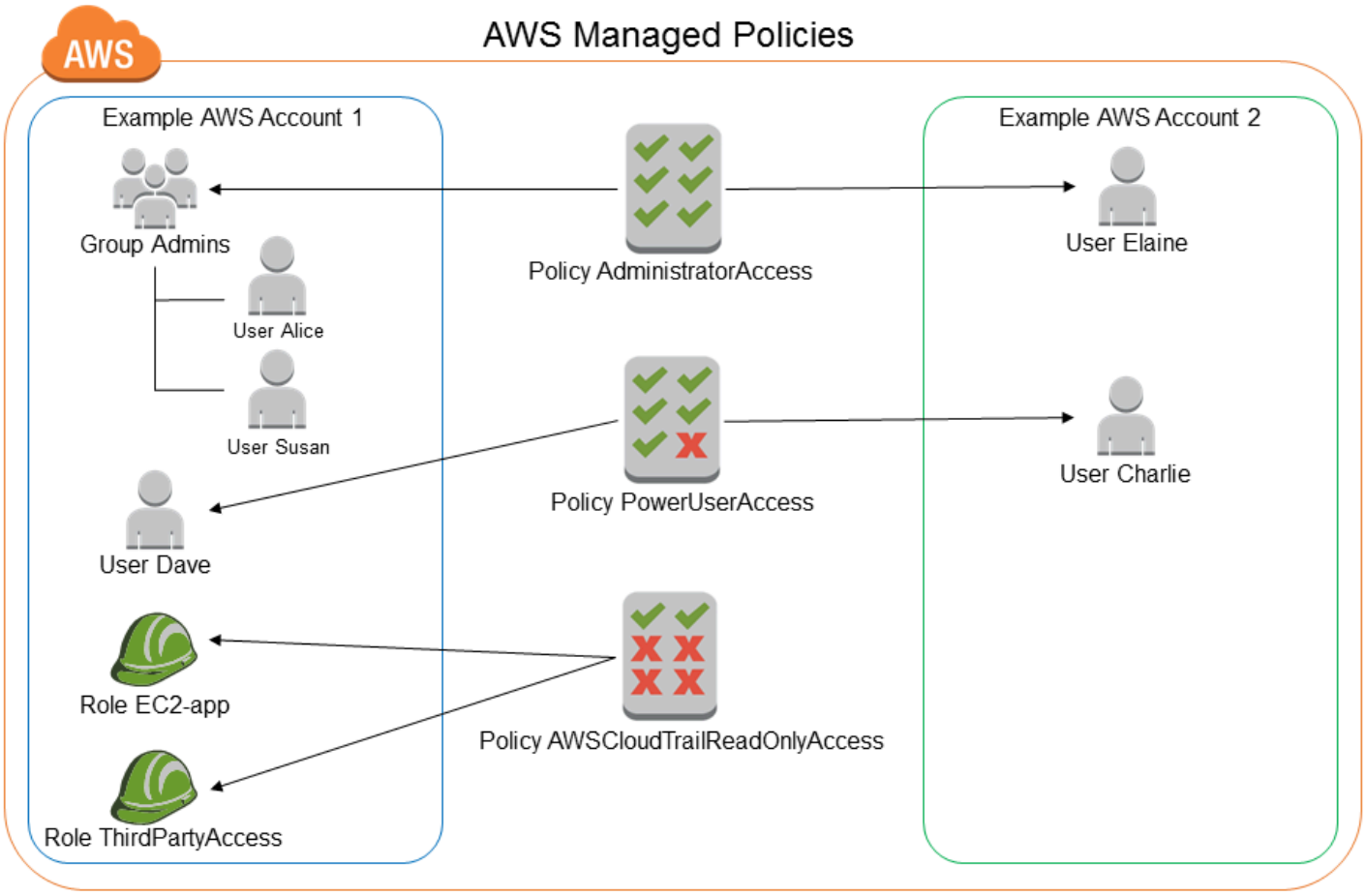
- AWSCodeCommitPowerUser
- AWSKeyManagementServicePowerUser

Partial-access AWS politiques gérées : elles fournissent des niveaux d'accès spécifiques Services AWS sans autoriser les autorisations de gestion des autorisations. En voici quelques exemples :

- AmazonMobileAnalyticsWriteOnlyAccess
- AmazonEC2ReadOnlyAccess

Politiques de AWS gestion des fonctions : ces politiques s'alignent étroitement sur les fonctions professionnelles couramment utilisées dans le secteur informatique et facilitent l'octroi d'autorisations pour ces fonctions. L'un des principaux avantages de l'utilisation des politiques relatives aux fonctions professionnelles est qu'elles sont maintenues et mises à jour au AWS fur et à mesure que de nouveaux services et opérations d'API sont introduits. Par exemple, la fonction de AdministratorAccess travail fournit un accès complet et une délégation d'autorisations à tous les services et ressources de AWS. Nous vous recommandons de n'utiliser cette politique que pour l'administrateur de compte. Pour les utilisateurs expérimentés qui ont besoin d'un accès complet à tous les services, à l'exception d'un accès limité à IAM et AWS Organizations, utilisez la fonction PowerUserAccess job. Pour obtenir la liste et la description des politiques de fonctions professionnelles, consultez la page AWS politiques gérées pour les fonctions professionnelles.

Le schéma suivant illustre les politiques AWS gérées. Le diagramme montre trois politiques AWS gérées : AdministratorAccess PowerUserAccess, et AWS CloudTrail _ ReadOnlyAccess. Notez qu'une seule politique AWS gérée peut être attachée à des entités principales dans différentes entités Comptes AWS, et à différentes entités principales dans une seule Compte AWS.



Politiques gérées par le client

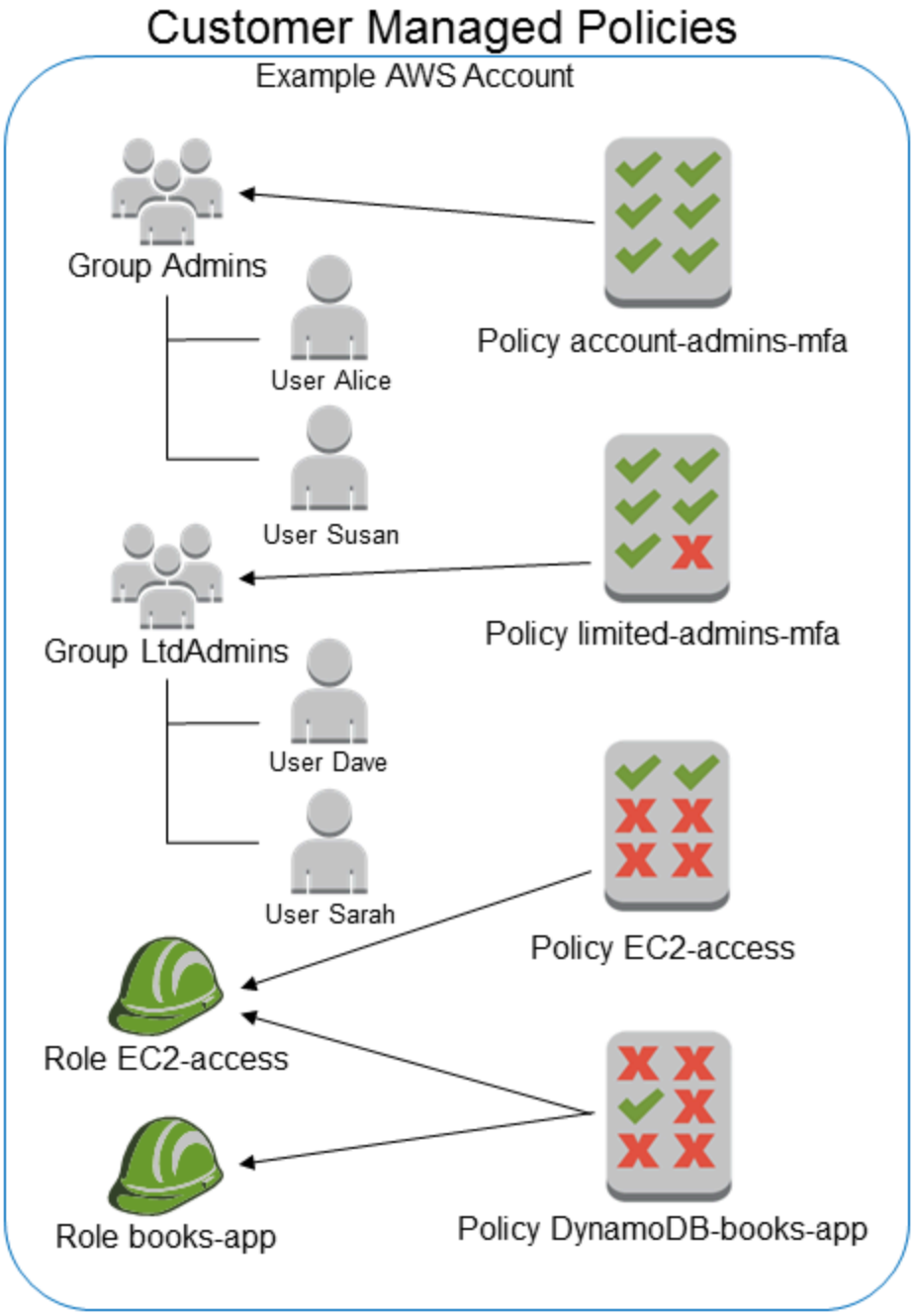
Vous pouvez créer vous-même des politiques autonomes Compte AWS que vous pouvez associer à des entités principales (utilisateurs IAM, groupes IAM et rôles IAM). Vous créez ces politiques gérées par le client pour vos cas d'utilisation spécifiques, et vous pouvez les modifier et les mettre à jour aussi souvent que vous le souhaitez. À l'instar des politiques AWS gérées, lorsque vous associez une politique à une entité principale, vous lui accordez les autorisations définies dans la stratégie. Lorsque vous mettez à jour les autorisations dans la politique, les changements sont appliqués à toutes les entités du principal auxquelles la politique est attachée.

Pour créer une politique gérée par le client, une bonne méthode consiste à commencer par copier une politique gérée par AWS . De cette façon, vous êtes sûr que la politique est correcte au départ ; il vous suffit ensuite de la personnaliser en fonction de votre environnement.

Le diagramme suivant illustre des politiques gérées par le client. Chaque politique est une entité dans IAM avec son propre Amazon Resource Name (ARN) dans lequel figure le nom de la politique. Notez

que la même politique peut être associée à plusieurs entités principales. Par exemple, la même DynamoDB-books-app politique est associée à deux rôles IAM différents.

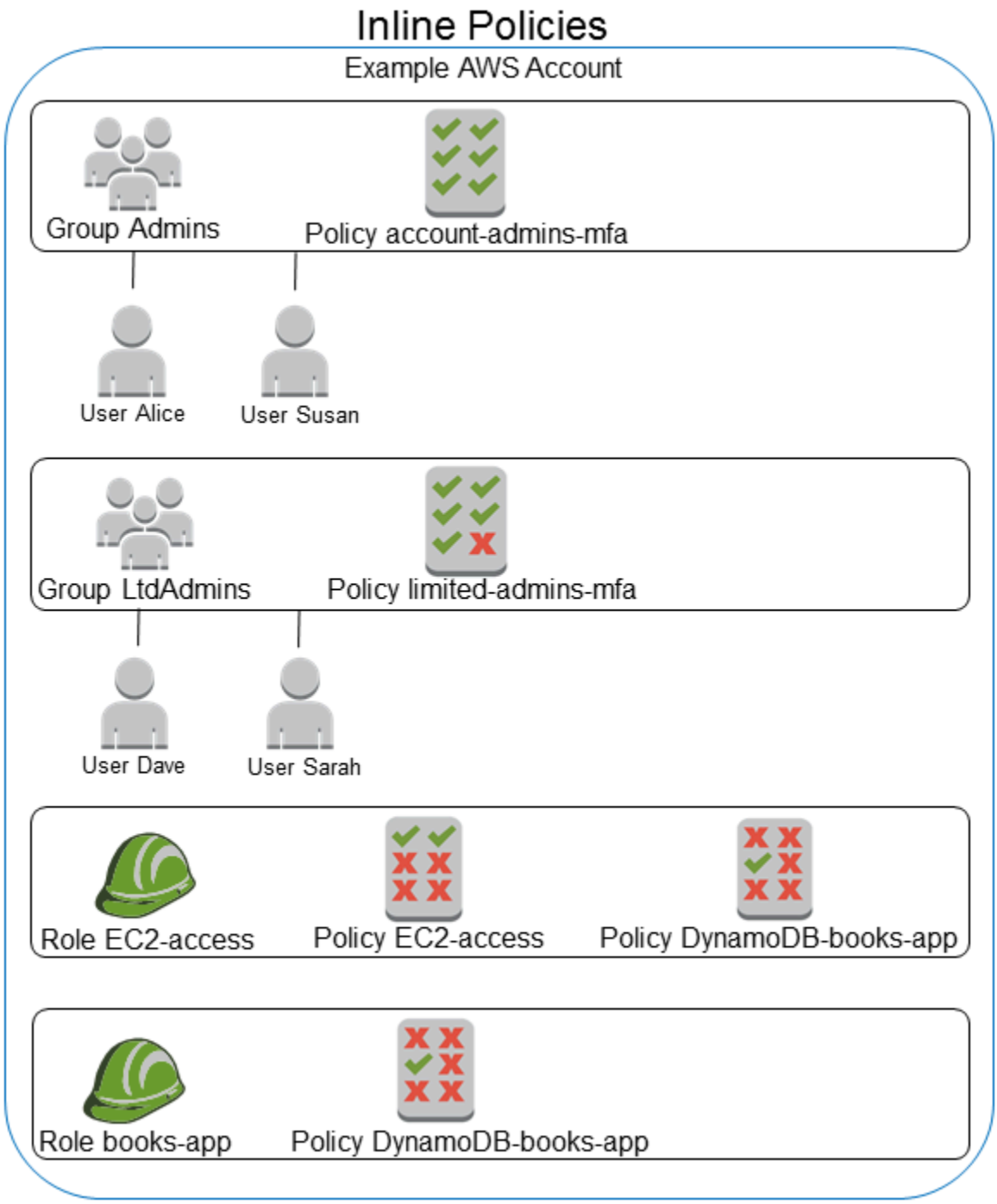
Pour de plus amples informations, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client.



Politiques en ligne

Une politique en ligne est une politique créée pour une identité IAM (utilisateur, groupe d’utilisateurs ou rôle). Les politiques en ligne maintiennent une relation un-à-un stricte entre une politique et une identité. Elles sont supprimées lorsque vous supprimez l'identité. Vous pouvez créer une politique et l'intégrer dans une identité, soit lors de la création de l'identité, soit ultérieurement. Si une politique peut s'appliquer à plusieurs entités, il est préférable d'utiliser une politique gérée.

Le diagramme suivant illustre des politiques en ligne. Chaque politique fait partie intégrante de l'utilisateur, du groupe ou du rôle. Notez que deux rôles incluent la même politique (la DynamoDB-books-app politique), mais qu'ils ne partagent pas une seule politique. Chaque rôle possède sa propre copie de la politique.



Choix entre politiques gérées et politiques en ligne

Prenez en compte vos cas d'utilisation lorsque vous devez choisir entre les politiques gérées et les politiques en ligne. Dans la plupart des cas, nous vous recommandons d'utiliser des politiques gérées plutôt que des politiques en ligne.

> **Note**
>
> Vous pouvez utiliser à la fois des politiques gérées et des politiques en ligne pour définir des autorisations communes et uniques pour une entité du principal.

Les politiques gérées fournissent les fonctions suivantes :

Réutilisation

Une même politique gérée peut être attachée à plusieurs entités du principal (utilisateurs, groupes et rôles). Vous pouvez créer une bibliothèque de politiques qui définissent des autorisations utiles pour vous Compte AWS, puis associer ces politiques aux entités principales selon les besoins.

Gestion centralisée des modifications

Lorsque vous modifiez une politique gérée, le changement est appliqué à toutes les entités du principal auxquelles la politique est attachée. Par exemple, si vous souhaitez ajouter une autorisation pour une nouvelle AWS API, vous pouvez mettre à jour une politique gérée par le client ou associer une politique AWS gérée pour ajouter l'autorisation. Si vous utilisez une politique AWS gérée, AWS mettez-la à jour. Lorsqu'une politique gérée est mise à jour, les modifications sont appliquées à toutes les entités du principal auxquelles la politique gérée est rattachée. En revanche, pour modifier une politique en ligne, vous devez modifier individuellement chaque identité contenant la politique en ligne. Par exemple, si un groupe et un rôle contiennent tous les deux la même politique en ligne, vous devez modifier individuellement les deux entités du principal pour modifier la politique.

Versioning et restauration

Lorsque vous modifiez une politique gérée par le client, la politique modifiée ne remplace pas la politique existante. À la place, IAM crée une nouvelle version de la politique gérée. IAM stocke jusqu'à cinq versions de vos politiques gérées par le client. Vous pouvez utiliser les versions de politique pour restaurer une version antérieure, si nécessaire.

> **Note**
>
> Une version de politique est différente d'un élément de politique `Version`. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. Pour en savoir plus sur les versions de politiques, consultez the section called

"Gestion des versions des politiques IAM". Pour en savoir plus sur l'élément de politique `Version`, consultez Éléments de politique JSON IAM : Version.

Délégation de la gestion des autorisations

Vous pouvez autoriser les utilisateurs de votre site Compte AWS à joindre et à détacher des politiques tout en gardant le contrôle sur les autorisations définies dans ces politiques. Pour ce faire, vous pouvez ainsi désigner certains utilisateurs en tant qu'administrateurs disposant de droits complets, autrement dit autorisés à créer, mettre à jour et supprimer des politiques. D'autres utilisateurs peuvent ensuite être désignés en tant qu'administrateurs limitées. Ces administrateurs restreints pouvent attacher des politiques à d'autres entités du principal, mais uniquement celles que vous leur avez autorisés à attacher.

Pour plus d'informations sur la délégation de la gestion des autorisations, consultez Contrôle de l'accès aux politiques.

Limites de caractères des politiques plus grandes

La limite maximale de caractères pour les politiques gérées est supérieure à la limite de caractères pour les politiques de groupe en ligne. Si vous atteignez la limite de taille de caractères pour la politique en ligne, vous pouvez créer d'autres groupes IAM et associer la politique gérée au groupe.

Pour plus d'informations sur les quotas et les limites, consultez IAM et AWS STS quotas.

Mises à jour automatiques pour les politiques AWS gérées

AWS gère les politiques AWS gérées et les met à jour lorsque cela est nécessaire, par exemple pour ajouter des autorisations pour de nouveaux AWS services, sans que vous ayez à apporter de modifications. Les mises à jour sont automatiquement appliquées aux principales entités auxquelles vous avez attaché la politique AWS gérée.

Premiers pas avec les politiques gérées

Nous vous recommandons d'utiliser des politiques qui accordent le moins de privilèges ou d'accorder uniquement les autorisations requises pour effectuer une tâche. Le moyen le plus sûr d'octroyer le moindre privilège consiste à écrire une politique gérée par le client contenant uniquement les autorisations requises par votre équipe. Vous devez créer un processus pour autoriser votre équipe à demander plus d'autorisations si nécessaire. Il faut du temps et de l'expertise pour créer des

politiques gérées par le client IAM qui ne fournissent à votre équipe que les autorisations dont elle a besoin.

Pour commencer à ajouter des autorisations à vos identités IAM (utilisateurs, groupes d'utilisateurs et rôles), vous pouvez utiliserAWS stratégies gérées. AWS les politiques gérées n'accordent pas les autorisations du moindre privilège. Vous devez prendre en compte le risque de sécurité constitué par l'octroi, à vos principaux, de davantage d'autorisations que nécessaire pour accomplir leur tâche.

Vous pouvez associer des politiques AWS gérées, y compris des fonctions de travail, à n'importe quelle identité IAM. Pour de plus amples informations, veuillez consulter Ajout et suppression d'autorisations basées sur l'identité IAM.

Pour passer aux autorisations du moindre privilège, vous pouvez exécuter Gestion des identités et des accès AWS Access Analyzer pour surveiller les principaux à l'aide de politiques AWS gérées. Lorsque vous savez quelles autorisations ils utilisent, vous pouvez écrire ou générer une politique gérée par le client avec uniquement les autorisations requises pour votre équipe. Cela est moins sûr, mais offre plus de flexibilité à mesure que vous apprenez comment votre équipe utilise AWS. Pour de plus amples informations, veuillez consulter Génération d'une politique IAM Access Analyzer.

AWS les politiques gérées sont conçues pour fournir des autorisations pour de nombreux cas d'utilisation courants. Pour plus d'informations sur les politiques AWS gérées conçues pour des fonctions professionnelles spécifiques, consultezAWS politiques gérées pour les fonctions professionnelles.

Pour obtenir la liste des politiques AWS gérées, consultez le Guide de référence des politiques AWS gérées.

Utilisation de politiques en ligne

Les politiques intégrées sont utiles si vous souhaitez maintenir une one-to-one relation stricte entre une stratégie et l'identité à laquelle elle est appliquée. Par exemple, si vous voulez éviter que les autorisations d'une politique ne soient accidentellement affectées à une identité autre que celle à laquelle elles sont destinées. Lorsque vous utilisez une politique en ligne, ses autorisations ne peuvent pas être attachées par inadvertance à la mauvaise identité. En outre, lorsque vous utilisez le Console de gestion AWS pour supprimer cette identité, les politiques intégrées à l'identité sont également supprimées car elles font partie de l'entité principale.

Conversion d’une politique en ligne en politique gérée

Si vous avez des politiques en ligne dans votre compte, vous pouvez les convertir en politiques gérées. Pour ce faire, copiez la politique dans une nouvelle politique gérée. Ensuite, attachez la nouvelle politique à l'identité qui inclut la politique en ligne. Ensuite, supprimez la politique en ligne.

Transformer une politique en ligne en politique gérée

Pour convertir une politique en ligne en politique gérée

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), Users (Utilisateurs) ou Roles (Rôles).
3. Choisissez dans la liste le nom du groupe d'utilisateurs, de l'utilisateur ou du rôle pour lequel vous souhaitez supprimer la politique.
4. Sélectionnez l’onglet Autorisations.
5. Pour les groupes IAM, sélectionnez le nom de la politique en ligne que vous souhaitez supprimer. Pour les utilisateurs et les rôles, choisissez Afficher ***n*** plus, si nécessaire, puis développez la politique intégrée que vous souhaitez supprimer.
6. Choisissez Copier pour copier le document de politique JSON pour la politique.
7. Dans le panneau de navigation, choisissez Politiques.
8. Choisissez Créer une politique, puis l'option JSON.
9. Remplacez le texte existant par votre texte de politique JSON, puis choisissez Suivant.
10. Saisissez un nom et une description facultative pour votre politique et choisissez Créer une politique.
11. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), Users (Utilisateurs) ou Roles (Rôles), puis sélectionnez le nom du groupe d'utilisateurs, de l'utilisateur ou du rôle pour lequel vous souhaitez supprimer la politique.
12. Choisissez l'onglet Autorisations, puis Ajouter des autorisations.
13. Pour les groupes IAM, cochez la case en regard du nom de votre nouvelle politique, choisissez Ajouter des autorisations, puis Attacher la politique. Pour les utilisateurs ou les rôles, choisissez Ajouter des autorisations. Sur la page suivante, choisissez Attacher directement les politiques existantes, cochez la case en regard du nom de votre nouvelle politique, choisissez Suivant, puis Ajouter des autorisations.

Vous êtes renvoyé à la page Summary (Résumé) de votre groupe d'utilisateurs, utilisateur ou rôle.

14. Cochez la case en regard de la politique en ligne que vous souhaitez supprimer et choisissez Supprimer.

Obsolète AWS stratégies gérées

Pour simplifier l'attribution des autorisations, AWS fournit des politiques gérées, c'est-à-dire des politiques prédéfinies prêtes à être associées à vos utilisateurs, groupes et rôles IAM.

Il est parfois AWS nécessaire d'ajouter une nouvelle autorisation à une politique existante, par exemple lors de l'introduction d'un nouveau service. L'ajout d'une nouvelle autorisation à une politique existante n'impacte aucune des fonctions, pas plus qu'elle ne les supprime.

Cependant, AWS vous pouvez choisir de créer une nouvelle politique lorsque les modifications nécessaires pourraient avoir un impact sur les clients si elles étaient appliquées à une politique existante. Par exemple, la suppression d'autorisations d'une politique existante peut affecter les autorisations d'une entité ou application IAM qui en dépend, ce qui risque de perturber une opération critique.

Par conséquent, lorsqu'un tel changement est requis, AWS crée une toute nouvelle politique avec les modifications requises et la met à la disposition des clients. L'ancienne politique est ensuite marquée comme étant obsolète. Pour plus d'informations, consultez la section Politiques AWS gérées obsolètes dans le AWS Managed Policy Reference Guide.

Mettez en place des barrières de protection des autorisations à l'aide de périmètres de données

Les barrières de périmètre de données sont destinées à servir de limites permanentes pour aider à protéger vos données sur un large éventail de comptes et de AWS ressources. Les périmètres de données suivent les pratiques exemplaires de sécurité IAM pour établir des garde-fous d'autorisation sur plusieurs comptes. Ces garde-fous relatifs aux autorisations à l'échelle de l'organisation ne remplacent pas vos contrôles d'accès précis existants. Ils fonctionnent plutôt comme des contrôles d'accès grossiers qui aident à améliorer votre politique en matière de sécurité, en garantissant que les utilisateurs, les rôles et les ressources respectent un ensemble de normes de sécurité définies.

Un périmètre de données est un ensemble de barrières d'autorisation dans votre AWS environnement qui permettent de s'assurer que seules vos identités fiables accèdent à des ressources fiables à partir des réseaux prévus.

- Identités fiables : les responsables (rôles IAM ou utilisateurs) de vos AWS comptes et AWS services agissent en votre nom.
- Ressources fiables : ressources détenues par vos AWS comptes ou par AWS des services agissant en votre nom.
- Réseaux attendus : vos centres de données sur site et vos clouds privés virtuels (VPC), ou réseaux de AWS services agissant en votre nom.

> **Note**
>
> Dans certains cas, il se peut que vous deviez étendre votre périmètre de données pour y inclure également l'accès par vos partenaires commerciaux de confiance. Vous devez prendre en compte tous les modèles d'accès aux données prévus lorsque vous créez une définition des identités de confiance, des ressources de confiance et des réseaux attendus propres à votre entreprise et à votre utilisation des Services AWS.

Les contrôles du périmètre des données doivent être traités comme tout autre contrôle de sécurité dans le cadre du programme de sécurité de l'information et de gestion des risques. Cela signifie que vous devez effectuer une analyse des menaces pour identifier les risques potentiels au sein de votre environnement cloud, puis, en fonction de vos propres critères d'acceptation des risques, sélectionner et mettre en œuvre des contrôles de périmètre de données appropriés. Pour mieux orienter l'approche itérative basée sur les risques en matière de mise en œuvre du périmètre des données, vous devez comprendre quels risques de sécurité et quels vecteurs de menace sont pris en compte par les contrôles du périmètre des données, ainsi que vos priorités en matière de sécurité.

Contrôles du périmètre de données

Les contrôles grossiers du périmètre des données vous aident à atteindre six objectifs de sécurité distincts sur trois périmètres de données grâce à la mise en œuvre de différentes combinaisons des Types de politique et des clés de condition.

Périmètre	Objectif de contrôle	Utilisation	Appliqué sur	Clé de contexte de condition globale
Identity	Seules les identités de confiance peuvent accéder à mes ressources	RCP	Ressources	aws : PrincipalOrg ID aws : PrincipalOrgPaths aws : PrincipalAccount aws : PrincipalIsAwsService aws : SourceOrgID aws : SourceOrgPath aws : SourceAccount
	Seules les identités de confiance sont autorisées à partir de mon réseau	Politique de point de terminaison d'un VPC	Réseau	
Ressources	Vos identités ne peuvent accéder qu’à des ressources de confiance	SCP	Identités	aws : ResourceOrg ID aws : ResourceOrgPaths aws : ResourceAccount
	Seules les ressources de confiance sont accessibles à partir de votre réseau	Politique de point de terminaison d'un VPC	Réseau	

Périmètre	Objectif de contrôle	Utilisation	Appliqué sur	Clé de contexte de condition globale
Réseau	Vos identités ne peuvent accéder aux ressources qu'à partir des réseaux attendus	SCP	Identités	aws : SourceIp aws : SourceVpc aws : SourceVpce
	Vos ressources ne sont accessibles qu'à partir des réseaux attendus	RCP	Ressources	aws : VpceAccount aws : VpceOrgPaths aws : VpceOrgID aws:ViaAWSService aws : PrincipalIsAwsService

Vous pouvez considérer les périmètres de données comme la création d'une frontière stricte autour de vos données afin d'empêcher les schémas d'accès involontaires. Bien que les périmètres de données puissent empêcher tout accès involontaire à grande échelle, vous devez tout de même prendre des décisions précises en matière de contrôle d'accès. L'établissement d'un périmètre de données ne diminue en rien la nécessité d'affiner en permanence les autorisations en utilisant des outils tels que l'analyseur d'accès IAM dans le cadre de votre parcours vers le principe du moindre privilège.

Pour appliquer des contrôles du périmètre des données aux ressources qui ne sont actuellement pas prises en charge par les RCP, vous pouvez utiliser des politiques basées sur les ressources étant directement associées aux ressources. Pour obtenir la liste des services compatibles RCP et les politiques basées sur les ressources, consultez Politiques de contrôle des ressources (RCP) et AWS services qui fonctionnent avec IAM.

Pour appliquer les contrôles périmétriques du réseau, nous vous recommandons d'utiliser `aws:VpceOrgID`, `aws:VpceOrgPaths` et `aws:VpceAccount` uniquement si tous les services auxquels vous souhaitez restreindre l'accès sont actuellement pris en charge. L'utilisation de ces clés de condition avec des services non pris en charge peut entraîner des résultats d'autorisation imprévus. Pour connaître la liste des services prenant en charge les clés, consultez AWS clés de contexte de condition globale. Si vous devez appliquer les contrôles à un plus large éventail de services, envisagez d'utiliser `aws:SourceVpc` et `aws:SourceVpce` à la place.

Périmètre d'identité

Un périmètre d'identité est un ensemble de contrôles d'accès préventifs grossiers qui permettent de s'assurer que seules les identités fiables peuvent accéder à vos ressources et que seules les identités fiables sont autorisées sur votre réseau. Les identités fiables incluent généralement les responsables (rôles ou utilisateurs) de vos AWS comptes et les AWS services agissant en votre nom. Toutes les autres identités sont considérées comme non fiables et sont bloquées par le périmètre d'identité, à moins qu'une exception explicite ne soit accordée.

Les clés de condition globales suivantes permettent de renforcer les contrôles du périmètre d'identité basés sur votre définition des identités de confiance. Utilisez ces clés dans les politiques de contrôle des ressources pour restreindre l'accès aux ressources, ou dans les politiques de point de terminaison d'un VPC pour restreindre l'accès à vos réseaux.

Identités que vous détenez

Vous pouvez utiliser les clés de condition suivantes pour définir les principaux IAM que vous créez et gérez dans votre. Comptes AWS

- aws : PrincipalOrg ID— Vous pouvez utiliser cette clé de condition pour vous assurer que les responsables IAM à l'origine de la demande appartiennent à l'organisation spécifiée dans. AWS Organizations
- aws : PrincipalOrgPaths— Vous pouvez utiliser cette clé de condition pour vous assurer que l'utilisateur IAM, le rôle IAM, le principal utilisateur AWS STS fédéré, le principal fédéré SAML, le principal fédéré OIDC ou pour Utilisateur racine d'un compte AWS faire la demande appartiennent à l'unité d'organisation (OU) spécifiée dans. AWS Organizations
- aws : PrincipalAccount— Vous pouvez utiliser cette clé de condition pour vous assurer que les ressources ne sont accessibles que par le compte principal que vous spécifiez dans la politique.

Identités de AWS services agissant en votre nom

Vous pouvez utiliser les clés de condition suivantes pour autoriser les AWS services à utiliser leur propre identité pour accéder à vos ressources lorsqu'ils agissent en votre nom.

- aws : PrincipalIsAWSServiceet aws : SourceOrg ID (ou aws : SourceOrgPaths etaws : SourceAccount) : vous pouvez utiliser ces clés de condition pour vous assurer que lorsque Service AWS les principaux utilisateurs accèdent à vos ressources, ils le font uniquement pour le compte d'une ressource de l'organisation, de l'unité organisationnelle ou d'un compte dans AWS Organizations.

Pour plus d'informations, consultez la section Établissement d'un périmètre de données sur AWS : Autoriser uniquement les identités fiables à accéder aux données de l'entreprise.

Périmètre de ressource

Un périmètre de ressources est un ensemble de contrôles d'accès préventifs grossiers qui permettent de s'assurer que vos identités ne peuvent accéder qu'à des ressources fiables et que seules les ressources fiables sont accessibles depuis votre réseau. Les ressources fiables incluent généralement les ressources détenues par vos AWS comptes ou par AWS des services agissant en votre nom.

Les clés de condition globales suivantes permettent de renforcer les contrôles du périmètre des ressources en fonction de votre définition des ressources de confiance. Utilisez ces clés dans les politiques de contrôle des services (SCP) pour restreindre les ressources auxquelles vos identités peuvent accéder, ou dans les politiques point de terminaison d'un VPC pour restreindre les ressources accessibles depuis vos réseaux.

Ressources qui vous appartiennent

Vous pouvez utiliser les clés de condition suivantes pour définir les AWS ressources que vous créez et gérez dans votre Comptes AWS.

- aws : ResourceOrg ID— Vous pouvez utiliser cette clé de condition pour vous assurer que la ressource à laquelle vous accédez appartient à l'organisation spécifiée dans AWS Organizations.
- aws : ResourceOrgPaths— Vous pouvez utiliser cette clé de condition pour vous assurer que la ressource à laquelle vous accédez appartient à l'unité organisationnelle (UO) spécifiée dans AWS Organizations.

- aws : ResourceAccount— Vous pouvez utiliser cette clé de condition pour vous assurer que la ressource à laquelle vous accédez appartient à la ressource spécifiée Compte AWS.

Ressources de AWS services agissant en votre nom

Dans certains cas, il se peut que vous deviez autoriser l'accès à des ressources qui vous AWS appartiennent, à des ressources qui n'appartiennent pas à votre organisation et qui sont accessibles par vos responsables ou par des AWS services agissant en votre nom. Pour plus d'informations sur ces scénarios, consultez la section Établissement d'un périmètre de données sur AWS : Autoriser uniquement les ressources fiables de mon organisation.

Périmètre de réseau

Un périmètre réseau est un ensemble de contrôles d'accès préventifs grossiers qui permettent de s'assurer que vos identités peuvent accéder aux ressources uniquement à partir des réseaux prévus et que vos ressources ne sont accessibles qu'à partir des réseaux prévus. Les réseaux attendus comprennent généralement vos centres de données sur site et vos clouds privés virtuels (VPC), ainsi que les réseaux de services AWS agissant en votre nom.

Les clés de condition globales suivantes permettent de renforcer les contrôles du périmètre de réseau en fonction de votre définition des réseaux attendus. Utilisez ces clés dans les politiques de contrôle des services (SCP) pour restreindre les réseaux à partir desquels vos identités peuvent communiquer, ou dans les politiques de contrôle des ressources (RCP) pour restreindre l'accès aux ressources aux réseaux attendus.

Les réseaux que vous détenez

Vous pouvez utiliser les clés de condition suivantes pour définir les réseaux que vos employés et applications sont censés utiliser pour accéder à vos ressources, telles que la plage CIDR IP de votre entreprise et vos VPC.

- aws : SourceIp— Vous pouvez utiliser cette clé de condition pour vous assurer que l'adresse IP du demandeur se situe dans une plage IP spécifiée.
- aws : SourceVpc— Vous pouvez utiliser cette clé de condition pour vous assurer que le point de terminaison du VPC par lequel passe la demande appartient au VPC spécifié.
- aws : SourceVpce— Vous pouvez utiliser cette clé de condition pour vous assurer que la demande passe par le point de terminaison VPC spécifié.
- aws : VpceAccount— Vous pouvez utiliser cette clé de condition pour vous assurer que les demandes passent par les points de terminaison VPC appartenant au compte spécifié AWS .

- aws : VpceOrgPaths— Vous pouvez utiliser cette clé de condition pour vous assurer que les demandes proviennent de points de terminaison VPC appartenant à des comptes appartenant à l'unité d'organisation (UO) spécifiée dans. AWS Organizations
- aws : VpceOrg ID— Vous pouvez utiliser cette clé de condition pour vous assurer que les demandes proviennent de points de terminaison VPC appartenant à des comptes de l'organisation spécifiée dans. AWS Organizations

`aws:VpceAccount`, `aws:VpceOrgPaths` et `aws:VpceOrgID` sont particulièrement utiles pour mettre en œuvre des contrôles périmétriques réseau qui se mettent automatiquement à l'échelle en fonction de l'utilisation de vos points de terminaison d'un VPC, sans nécessiter de mise à jour des politiques lorsque vous créez de nouveaux points de terminaison. Pour connaître la liste des Services AWS qui prennent en charge ces clés, consultez AWS clés de contexte de condition globale.

Réseaux de AWS services agissant en votre nom

Vous pouvez utiliser les touches de condition suivantes pour autoriser les AWS services à accéder à vos ressources depuis leurs réseaux lorsqu'ils agissent en votre nom.

- aws:ViaAWSService— Vous pouvez utiliser cette clé de condition pour vous assurer que vous Services AWS pouvez effectuer des demandes au nom de votre mandant en utilisant Transmission des sessions d'accès (FAS).
- aws : PrincipalIsAWSService— Vous pouvez utiliser cette clé de condition pour vous assurer que vous Services AWS pouvez accéder à vos ressources en utilisantAWS principes de service.

Il existe d'autres scénarios dans lesquels vous devez autoriser l'accès aux Services AWS qui accèdent à vos ressources depuis l'extérieur de votre réseau. Pour plus d'informations, consultez la section Établissement d'un périmètre de données sur AWS : Autoriser l'accès aux données de l'entreprise uniquement à partir des réseaux prévus.

Ressources utiles sur les périmètres de données

Les ressources suivantes peuvent vous aider à en savoir plus sur les périmètres de données au sein d' AWS.

- Périmètres de données activés AWS : découvrez les périmètres de données, leurs avantages et leurs cas d'utilisation.

- Série d'articles de blog : Établissement d'un périmètre de données sur AWS — Ces articles de blog fournissent des conseils prescriptifs sur l'établissement de votre périmètre de données à grande échelle, y compris des considérations clés en matière de sécurité et de mise en œuvre.
- Exemples de politiques de périmètre de données : ce GitHub référentiel contient des exemples de politiques qui couvrent certains modèles courants pour vous aider à implémenter un périmètre de données AWS.
- Data Perimeter Helper : cet outil vous aide à concevoir et à anticiper l'impact de vos contrôles de périmètre de données en analysant les activités d'accès dans vos journaux AWS CloudTrail.

Limites d'autorisations pour les entités IAM

AWS prend en charge les limites d'autorisations pour les entités IAM (utilisateurs ou rôles). Une limite d'autorisations est une fonctionnalité avancée permettant d'utiliser une politique gérée pour définir les autorisations maximales qu'une politique basée sur l'identité peut accorder à une entité IAM. La limite d'autorisations d'une entité lui permet d'effectuer uniquement les actions autorisées à la fois par ses politiques basées sur l'identité et ses limites d'autorisations.

Pour de plus amples informations sur les types de politiques, veuillez consulter Types de politique.

> **Important**
>
> N'utilisez pas de déclarations de politique basée sur les ressources qui incluent un élément de politique `NotPrincipal` ayant un effet `Deny` sur les utilisateurs IAM ou les rôles auxquels est attachée une politique de limite des autorisations. L'élément `NotPrincipal` ayant un effet `Deny` refusera toujours tout principal IAM auquel est attachée une politique de limite des autorisations, quelles que soient les valeurs spécifiées dans l'élément `NotPrincipal`. Certains utilisateurs ou rôles IAM qui auraient autrement eu accès à la ressource n'y ont donc plus accès.
>
> Nous vous recommandons de modifier vos déclarations de politique basée sur les ressources afin d'utiliser l'opérateur de condition `ArnNotEquals` avec la clé de contexte `aws:PrincipalArn` dans le but de limiter l'accès plutôt que l'élément `NotPrincipal`. Pour en savoir plus sur l'élément `NotPrincipal`, veuillez consulter AWS Éléments de politique JSON : NotPrincipal.

Vous pouvez utiliser une politique AWS gérée ou une politique gérée par le client pour définir les limites d'une entité IAM (utilisateur ou rôle). Cette politique limite le nombre maximum d'autorisations pour l'utilisateur ou le rôle.

Supposons, par exemple, que l'utilisateur IAM nommé `Shirley` soit autorisé à gérer uniquement Amazon S3 CloudWatch, Amazon et Amazon EC2. Pour appliquer cette règle, vous pouvez utiliser la politique suivante afin de définir les autorisations pour l'utilisateur `Shirley` :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*"
            ],
            "Resource": "*"
        }
    ]
}
```

Lorsque vous utilisez une politique pour définir la limite d'autorisations d'un utilisateur, elle limite les autorisations de l'utilisateur mais ne fournit pas seule les autorisations. Dans cet exemple, la politique définit les autorisations maximales pour toutes `Shirley` les opérations dans Amazon S3 et Amazon EC2. CloudWatch Shirley ne peut pas exécuter d'opérations dans les autres services, y compris IAM, même si elle dispose d'une politique d'autorisations qui le permet. Par exemple, vous pouvez ajouter la politique suivante à l'utilisateur `Shirley` :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
```

```
    "Resource": "*"
  }
}
```

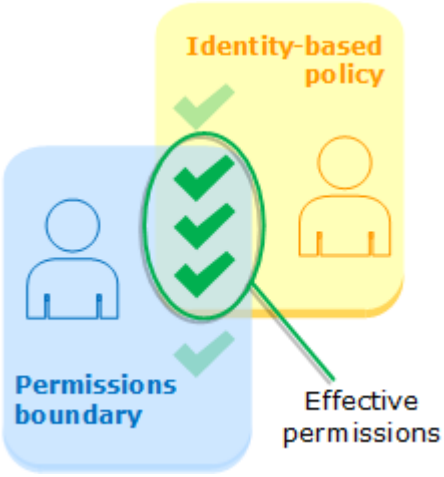
Cette politique permet de créer un utilisateur dans IAM. Si vous attachez cette politique d'autorisation à l'utilisateur `Shirley` et que Shirley tente de créer un utilisateur, l'opération échoue. Elle échoue car la limite des autorisations n'autorise pas l'opération `iam:CreateUser`. Compte tenu de ces deux politiques, Shirley n'est pas autorisée à effectuer des opérations dans AWS. Vous devez ajouter une politique d'autorisations différente pour autoriser des actions dans d'autres services, tels qu'Amazon S3. Sinon, vous pouvez mettre à jour la limite d'autorisations pour lui permettre de créer un utilisateur dans IAM.

Évaluation des autorisations effectives avec limites

La limite d'autorisations pour une entité IAM (utilisateur ou rôle) définit le nombre maximum d'autorisations maximum dont peut disposer une entité. Cela peut modifier les autorisations effectives pour cet utilisateur ou rôle. Les autorisations effectives d'une entité correspondent aux autorisations accordées par toutes les politiques affectant l'utilisateur ou le rôle. Dans un compte, les autorisations d'une entité peuvent être affectées par des stratégies basées sur l'identité, des stratégies basées sur les ressources, des limites d'autorisations, des stratégies de contrôle de service AWS Organizations ou des stratégies de session. Pour de plus amples informations sur les différents types de politiques, veuillez consulter Politiques et autorisations dans Gestion des identités et des accès AWS.

Si l'un de ces types de politique refuse explicitement l'accès pour une opération, la demande est refusée. Les autorisations accordées à une entité par plusieurs types d'autorisations sont plus complexes. Pour plus de détails sur la manière dont AWS les politiques sont évaluées, consultezLogique d'évaluation de politiques.

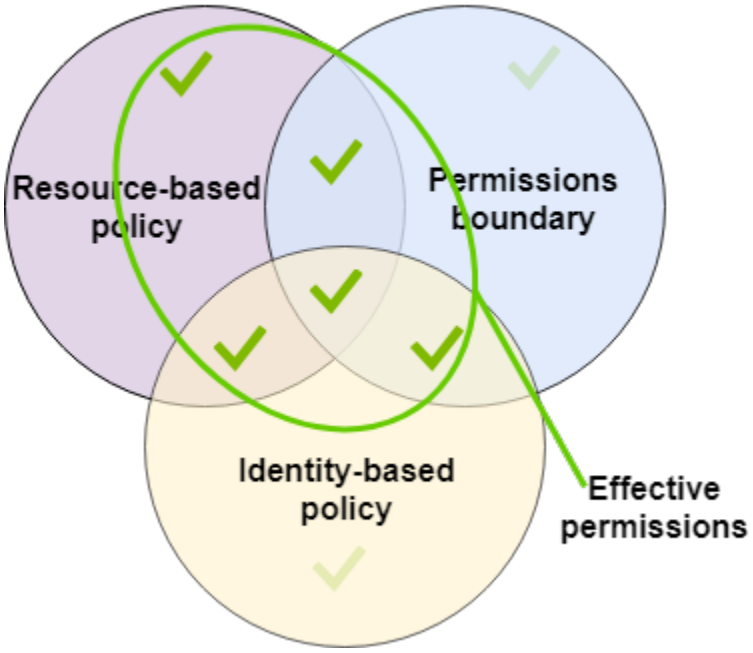
Identity-based politiques avec limites : les Identity-based politiques sont des politiques intégrées ou gérées qui sont associées à un utilisateur, à un groupe d'utilisateurs ou à un rôle. Identity-based les politiques accordent des autorisations à l'entité, et les limites des autorisations limitent ces autorisations. Les autorisations effectives sont une combinaison de ces deux types de politique. Un refus explicite dans l'une ou l'autre de ces stratégies remplace l'autorisation.



Resource-based politiques : Resource-based les politiques contrôlent la manière dont le principal spécifié peut accéder à la ressource à laquelle la politique est attachée.

Resource-based politiques pour les utilisateurs IAM

Au sein d'un même compte, les politiques basées sur les ressources qui accordent des autorisations à l'ARN d'un utilisateur IAM (qui n'est pas une session principale d'utilisateur AWS STS fédérée) ne sont pas limitées par un refus implicite dans une politique basée sur l'identité ou une limite d'autorisations.



Resource-based politiques relatives aux rôles IAM

Rôle IAM : Resource-based les politiques qui accordent des autorisations à un ARN de rôle IAM sont limitées par un refus implicite dans une limite d'autorisations ou une politique de session.

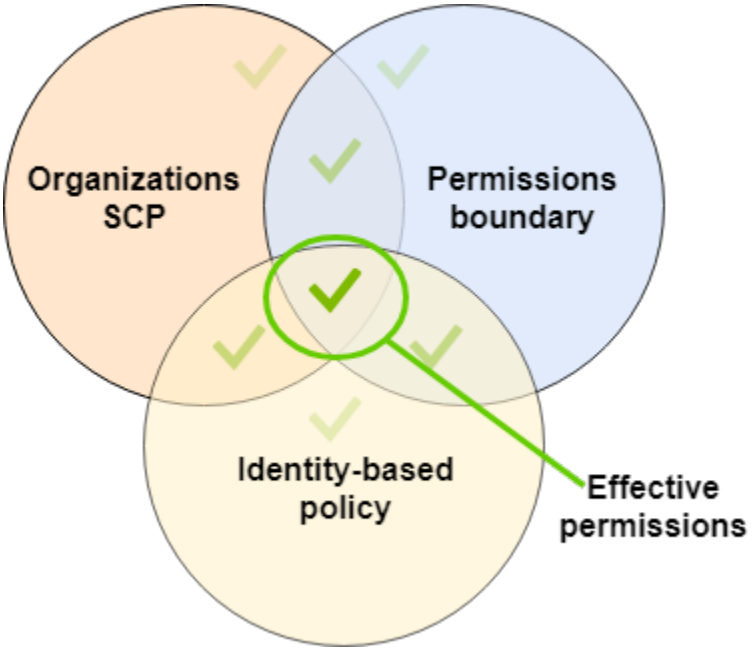
Séance de rôle IAM – Au sein d'un même compte, les politiques basées sur les ressources qui accordent des autorisations à un ARN de séance de rôle IAM accordent des autorisations directement à la séance de rôle endossée. Les autorisations accordées directement à une séance ne sont pas limitées par un rejet implicite dans une politique basée sur l'identité, une limite d'autorisations ou une politique de séance. Lorsque vous endossez un rôle et faites une demande, le principal qui fait la demande est l'ARN de séance du rôle IAM et non l'ARN du rôle lui-même.

Resource-based politiques relatives aux sessions principales AWS STS des utilisateurs fédérés

AWS STS sessions d'utilisateur principal fédérées : une session principale d'utilisateur AWS STS fédéré est une session créée par un appel. `GetFederationToken` Lorsqu'un utilisateur fédéré fait une demande, le principal qui effectue la demande est l'ARN d'utilisateur fédéré et non l'ARN de l'utilisateur IAM qui a fédéré. Dans le même compte, les politiques basées sur les ressources accordant des autorisations à un ARN d'utilisateur fédéré accordent des autorisations directement à la séance. Les autorisations accordées directement à une séance ne sont pas limitées par un rejet implicite dans une politique basée sur l'identité, une limite d'autorisations ou une politique de séance.

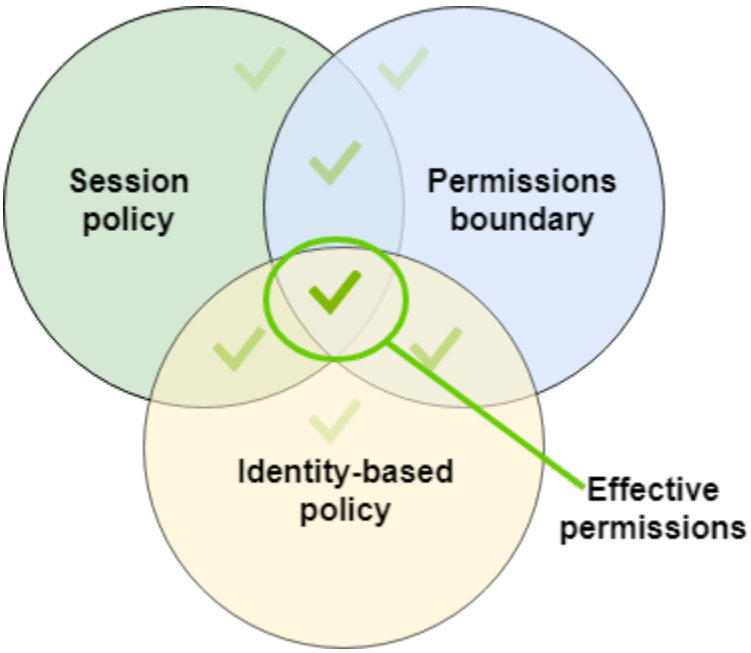
Toutefois, si une politique basée sur les ressources accorde une autorisation à l'ARN de l'utilisateur IAM qui a fédéré, les demandes effectuées par l'utilisateur principal AWS STS fédéré au cours de la session sont limitées par un refus implicite dans une limite d'autorisation ou une politique de session.

AWS Organizations SCP — Les SCP sont appliqués à un ensemble. Compte AWS Elles limitent les autorisations pour chaque demande faite par un principal dans le compte. Une entité IAM (utilisateur ou rôle) peut effectuer une demande qui est soumise à une politique de contrôle de service, une limite d'autorisations et une politique basée sur l'identité. Dans ce cas, la demande est autorisée uniquement si les trois types de politique l'autorisent. Les autorisations effectives sont la combinaison des trois types de politique. Un refus explicite dans l’une de ces politiques annule l’autorisation.



Vous pouvez savoir si votre compte est membre d'une organisation dans AWS Organizations. Les membres de l'organisation pourraient être affectés par une SCP. Pour afficher ces données à l'aide de la AWS CLI commande ou de AWS l'opération d'API, vous devez disposer des autorisations nécessaires à l'`organizations:DescribeOrganization`action pour votre AWS Organizations entité. Vous devez disposer d'autorisations supplémentaires pour effectuer l'opération dans la AWS Organizations console. Pour savoir si un SCP refuse l'accès à une demande spécifique ou pour modifier vos autorisations effectives, contactez votre AWS Organizations administrateur.

Politiques de session : les politiques de session sont des politiques avancées que vous passez en tant que paramètre lorsque vous programmez afin de créer une session temporaire pour un rôle ou un utilisateur fédéré. Les autorisations pour une session proviennent de l'entité IAM (utilisateur ou rôle) utilisée pour créer la session et de la politique de la session. Les autorisations de politique basée sur l'identité de l'entité sont limitées par la politique de la session et la limite d'autorisations. Les autorisations effectives pour cet ensemble de types de politique sont la combinaison des trois types de politique. Un refus explicite dans l'une de ces politiques annule l'autorisation. Pour de plus amples informations sur les politiques de session, veuillez consulter Politiques de session.



Délégation de responsabilité à d'autres utilisateurs à l'aide des limites d'autorisations

Vous pouvez utiliser des limites d'autorisations pour déléguer à des utilisateurs IAM de votre compte des tâches de gestion d'autorisations, comme la création d'utilisateurs,. Cela permet à d'autres utilisateurs d'exécuter des tâches en votre nom dans une limite d'autorisations spécifique.

Supposons, par exemple, que María soit l'administratrice du X-Company Compte AWS. Elle souhaite déléguer des tâches de création d'utilisateurs à Zhang. Toutefois, elle doit s'assurer que Zhang crée des utilisateurs qui respectent les règles de l'entreprise suivantes :

- Les utilisateurs ne peuvent pas utiliser IAM pour créer ou gérer des utilisateurs, groupes, rôles ou politiques.
- Les utilisateurs se voient refuser l'accès au compartiment `logs` Amazon S3 et ne peuvent pas accéder à l'instance Amazon EC2 `i-1234567890abcdef0`.
- Les utilisateurs ne peuvent pas supprimer leurs propres politiques de limite.

Pour appliquer ces règles, María exécute les tâches suivantes dont les détails sont inclus ci-dessous :

1. María crée la politique gérée `XCompanyBoundaries` pour utiliser en tant que limite d'autorisations pour tous les nouveaux utilisateurs du compte.

2. María crée la politique gérée `DelegatedUserBoundary` et l'assigne en tant que limite d'autorisations pour Zhang. Maria prend note de l'ARN de son administrateur et l'utilise dans la stratégie pour empêcher Zhang d'y accéder.
3. María crée la politique gérée `DelegatedUserPermissions` et l'attache en tant que limite d'autorisations pour Zhang.
4. María explique à Zhang ses nouvelles responsabilités et limites.

Tâche 1 : María doit d'abord créer une politique gérée pour définir la limite pour les nouveaux utilisateurs. María autorise Zhang à donner aux utilisateurs les politiques d'autorisations dont ils ont besoin, mais elle souhaite que ces derniers soient limités. Pour ce faire, elle crée la politique gérée par le client suivante avec le nom `XCompanyBoundaries`. Cette politique effectue les opérations suivantes :

- Permet aux utilisateurs d'avoir un accès complet à plusieurs services
- Autorise un accès autonome limité dans la console IAM. Cela signifie qu'ils peuvent changer leur mot de passe après s'être connecté à la console. Ils ne peuvent pas définir leur mot de passe initial. Pour que cela soit possible, ajoutez l'action `"*LoginProfile"` à l'instruction `AllowManageOwnPasswordAndAccessKeys`.
- Refuse aux utilisateurs l'accès au compartiment des journaux Amazon S3 ou à l'instance Amazon EC2 `i-1234567890abcdef0`

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ServiceBoundaries",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*",
                "dynamodb:*"
            ],
            "Resource": "*"
        },
```

```
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
        }
    ]
}
```

Chaque instruction répond à un objectif distinct :

1. L'`ServiceBoundaries`énoncé de cette politique permet un accès complet aux AWS services spécifiés. Cela signifie que de nouvelles actions de l'utilisateur dans ces services sont uniquement limitées par les politiques d'autorisations qui lui sont attachées.
2. L'instruction `AllowIAMConsoleForCredentials` autorise l'accès pour répertorier tous les utilisateurs IAM. Cet accès est nécessaire pour accéder à la page Utilisateurs dans la Console de gestion AWS. Il permet également d'afficher les exigences de mot de passe pour le compte, qui sont nécessaires lorsque vous modifiez votre mot de passe.
3. La déclaration `AllowManageOwnPasswordAndAccessKeys` autorise les utilisateurs à gérer uniquement leur propre mot de passe de console et les clés d'accès par programmation. Ceci est important si Zhang ou un autre administrateur attribue à un nouvel utilisateur une politique d'autorisations avec un accès complet à IAM. Dans ce cas, cet utilisateur peut ensuite modifier ses propres autorisations ou celles des autres utilisateurs. Cette instruction évite ce genre de problème.
4. L'instruction `DenyS3Logs` refuse explicitement l'accès au compartiment `logs`.
5. L'instruction `DenyEC2Production` refuse explicitement l'accès à l'instance `i-1234567890abcdef0`.

Tâche 2 : María souhaite autoriser Zhang à créer tous les X-Company utilisateurs, mais uniquement avec la limite `XCompanyBoundaries` des autorisations. Elle crée la politique gérée par le client suivante et la nomme `DelegatedUserBoundary`. Cette politique définit le nombre maximum de permissions dont dispose Zhang.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOrChangeOnlyWithBoundary",
            "Effect": "Allow",
            "Action": [
                "iam:AttachUserPolicy",
                "iam:CreateUser",
                "iam:DeleteUserPolicy",
                "iam:DetachUserPolicy",
                "iam:PutUserPermissionsBoundary",
                "iam:PutUserPolicy"
```

```
            ],
            "Resource": "*",
            "Condition": {
               "StringEquals": {
                 "iam:PermissionsBoundary": "arn:aws:iam::123456789012:policy/
XCompanyBoundaries"
                }
            }
        },
        {
            "Sid": "CloudWatchAndOtherIAMTasks",
            "Effect": "Allow",
            "Action": [
              "cloudwatch:*",
              "iam:CreateAccessKey",
              "iam:CreateGroup",
              "iam:CreateLoginProfile",
              "iam:CreatePolicy",
              "iam:DeleteGroup",
              "iam:DeletePolicy",
              "iam:DeletePolicyVersion",
              "iam:DeleteUser",
              "iam:GetAccountPasswordPolicy",
              "iam:GetGroup",
              "iam:GetLoginProfile",
              "iam:GetPolicy",
              "iam:GetPolicyVersion",
              "iam:GetRolePolicy",
              "iam:GetUser",
              "iam:GetUserPolicy",
              "iam:ListAccessKeys",
              "iam:ListAttachedRolePolicies",
              "iam:ListAttachedUserPolicies",
              "iam:ListEntitiesForPolicy",
              "iam:ListGroups",
              "iam:ListGroupsForUser",
              "iam:ListMFADevices",
              "iam:ListPolicies",
              "iam:ListPolicyVersions",
              "iam:ListRolePolicies",
              "iam:ListSSHPublicKeys",
              "iam:ListServiceSpecificCredentials",
              "iam:ListSigningCertificates",
              "iam:ListUserPolicies",
```

```
                "iam:ListUsers",
                "iam:SetDefaultPolicyVersion",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy",
                "iam:UpdateGroup",
                "iam:UpdateLoginProfile",
                "iam:UpdateUser"
            ],
            "NotResource": "arn:aws:iam::123456789012:user/Maria"
        },
        {
            "Sid": "NoBoundaryPolicyEdit",
            "Effect": "Deny",
            "Action": [
                "iam:CreatePolicyVersion",
                "iam:DeletePolicy",
                "iam:DeletePolicyVersion",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": [
                "arn:aws:iam::123456789012:policy/XCompanyBoundaries",
                "arn:aws:iam::123456789012:policy/DelegatedUserBoundary"
            ]
        },
        {
            "Sid": "NoBoundaryUserDelete",
            "Effect": "Deny",
            "Action": "iam:DeleteUserPermissionsBoundary",
            "Resource": "*"
        }
    ]
}
```

Chaque instruction répond à un objectif distinct :

1. L'instruction `CreateOrChangeOnlyWithBoundary` autorise Zhang à créer des utilisateurs IAM, mais uniquement s'il utilise la politique `XCompanyBoundaries` permettant de définir la limite d'autorisations. Cette instruction lui permet également de définir la limite d'autorisations pour les utilisateurs existants, mais uniquement à l'aide de cette même politique. Enfin, cette instruction autorise Zhang à gérer des politiques d'autorisations pour les utilisateurs avec cette limite d'autorisations définie.

2. L'instruction `CloudWatchAndOtherIAMTasks` autorise Zhang à exécuter les tâches de gestion d'autres utilisateurs, groupes et politiques. Il possède les autorisations de réinitialiser les mots de passe et de créer des clés d'accès pour tout utilisateur IAM non répertorié dans l'élément de politique `NotResource`. Cela lui permet d'aider les utilisateurs qui rencontrent des problèmes de connexion.
3. L'instruction `NoBoundaryPolicyEdit` refuse l'accès à Zhang pour mettre à jour la politique `XCompanyBoundaries`. Il n'est pas autorisé à modifier une politique utilisée pour définir la limite d'autorisations pour lui-même ou d'autres utilisateurs.
4. L'instruction `NoBoundaryUserDelete` interdit à Zhang de supprimer la limite d'autorisations pour lui-même ou d'autres utilisateurs.

María attribue ensuite la politique `DelegatedUserBoundary` comme limite d'autorisations pour l'utilisateur `Zhang`.

Tâche 3 : María doit créer une politique d'autorisations pour Zhang, car la limite d'autorisations restreint le nombre maximum d'autorisations, mais n'accorde pas à elle seule l'accès. Elle crée la politique suivante et la nomme `DelegatedUserPermissions`. Cette politique définit les opérations que Zhang peut exécuter au sein de la limite définie.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "IAM",
            "Effect": "Allow",
            "Action": "iam:*",
            "Resource": "*"
        },
        {
            "Sid": "CloudWatchLimited",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:GetDashboard",
                "cloudwatch:GetMetricData",
                "cloudwatch:ListDashboards",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics"
```

```
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3BucketContents",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::ZhangBucket"
        }
    ]
}
```

Chaque instruction répond à un objectif distinct :

1. L'instruction `IAM` de la politique accorde à Zhang un accès complet à IAM. Toutefois, ses autorisations IAM effectives sont uniquement restreintes par sa limite d'autorisations, car cette dernière autorise uniquement certaines opérations IAM.
2. La `CloudWatchLimited` déclaration permet à Zhang d'effectuer cinq actions dans CloudWatch. Ses limites d'autorisations autorisent toutes les actions CloudWatch, de sorte que ses CloudWatch autorisations effectives ne sont limitées que par sa politique d'autorisations.
3. L'instruction `S3BucketContents` autorise Zhang à répertorier le compartiment Amazon S3 `ZhangBucket`. Toutefois, sa limite d'autorisations n'autorise aucune action Amazon S3. Par conséquent, il ne peut pas exécuter d'opérations S3, quelle que soit sa politique d'autorisations.

> **Note**
>
> Les politiques de Zhang lui permettent de créer un utilisateur qui peut alors accéder à des ressources Amazon S3 auxquelles il ne peut pas accéder. En déléguant ces actions administratives, Maria approuve l'accès de Zhang à Amazon S3.

María assigne ensuite la politique `DelegatedUserPermissions` en tant que politique d'autorisations pour l'utilisateur `Zhang`.

Tâche 4 : Elle explique à Zhang comment créer un nouvel utilisateur. Elle lui indique qu'il peut créer de nouveaux utilisateurs avec les autorisations dont il a besoin, mais il doit leur assigner la politique `XCompanyBoundaries` en tant que limite d'autorisations.

Zhang exécute les tâches suivantes :

1. Zhang crée un utilisateur avec le Console de gestion AWS. Il saisit le nom d'utilisateur `Nikhil` et accorde à l'utilisateur l'accès à la console. Il décoche la case en regard de Requires password reset (Réinitialisation du mot de passe requise), car les politiques ci-dessus permettent à Zhang de modifier son mot de passe uniquement après s'être connecté à la console IAM.
2. Sur la page Définir les autorisations, Zhang choisit les politiques IAMFullAccess d' AmazonS3ReadOnlyAccessautorisations qui permettent à Nikhil de faire son travail.
3. Zhang ignore la section Set permissions boundary (Définir une limite d'autorisations), oubliant ainsi les instructions de María.
4. Zhang examine les détails de l'utilisateur et choisit Créer un utilisateur.

   L'opération échoue et l'accès est refusé. La limite d'autorisations `DelegatedUserBoundary` de Zhang exige que tous les utilisateurs qu'il crée utilisent la politique `XCompanyBoundaries` en tant que limite d'autorisations.
5. Zhang revient à la page précédente. Dans la section Set permissions boundary (Définir une limite d'autorisations), il choisit la politique `XCompanyBoundaries`.
6. Zhang examine les détails de l'utilisateur et choisit Créer un utilisateur.

   L'utilisateur est créé.

Lorsque Nikhil se connecte, il accède à IAM et Amazon S3, à l'exception des opérations refusées par la limite d'autorisations. Par exemple, il peut modifier son propre mot de passe dans IAM, mais ne peut pas créer d'autre utilisateur ou modifier ses politiques. Nikhil a un accès en lecture seule à Amazon S3.

Si une personne ajoute au compartiment `logs` une politique basée sur les ressources qui autorise Nikhil à placer un objet dans le compartiment, il ne peut toujours pas accéder au compartiment. En effet, toutes les actions sur le compartiment `logs` sont explicitement refusées par sa limite d'autorisations. Un refus explicite dans n'importe quel type de politique se traduit par le refus d'une demande.

Toutefois, si une politique basée sur les ressources attachée à un secret Secrets Manager permet à Nikhil d'effectuer l'action `secretsmanager:GetSecretValue`, Nikhil peut récupérer et déchiffrer le secret. En effet, les opérations Secrets Manager ne sont pas explicitement refusées par sa limite d'autorisations et les conditions de refus implicite dans les limites d'autorisations ne limitent pas les politiques basées sur les ressources.

Identity-based politiques et politiques basées sur les ressources

Une politique est un objet AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit leurs autorisations. Lorsque vous créez une politique d'autorisations pour limiter l'accès à une ressource, vous pouvez choisir une politique basée sur l'identité ou une politique basée sur les ressources.

Identity-based les politiques sont associées à un utilisateur, un groupe ou un rôle IAM. Ces politiques vous permettent de spécifier ce que peut faire cette identité (ses autorisations). Par exemple, vous pouvez attacher la politique à l'utilisateur IAM nommé John, en indiquant qu'il est autorisé à effectuer l'action Amazon EC2 `RunInstances`. La politique peut aussi indiquer que John est autorisé à extraire des éléments d'une table Amazon DynamoDB nommée `MyCompany`. Vous pouvez également autoriser John à gérer ses propres identifiants de sécurité IAM. Identity-based les politiques peuvent être gérées ou intégrées.

Resource-based les politiques sont associées à une ressource. Par exemple, vous pouvez associer des politiques basées sur les ressources aux compartiments Amazon S3, aux files d'attente Amazon SQS, aux points de terminaison VPC, aux clés de AWS Key Management Service chiffrement, aux tables et aux flux Amazon DynamoDB, ainsi qu'aux ressources. AWS Sign-In Pour obtenir la liste des services qui prennent en charge les politiques basées sur une ressource, consultez AWS services qui fonctionnent avec IAM.

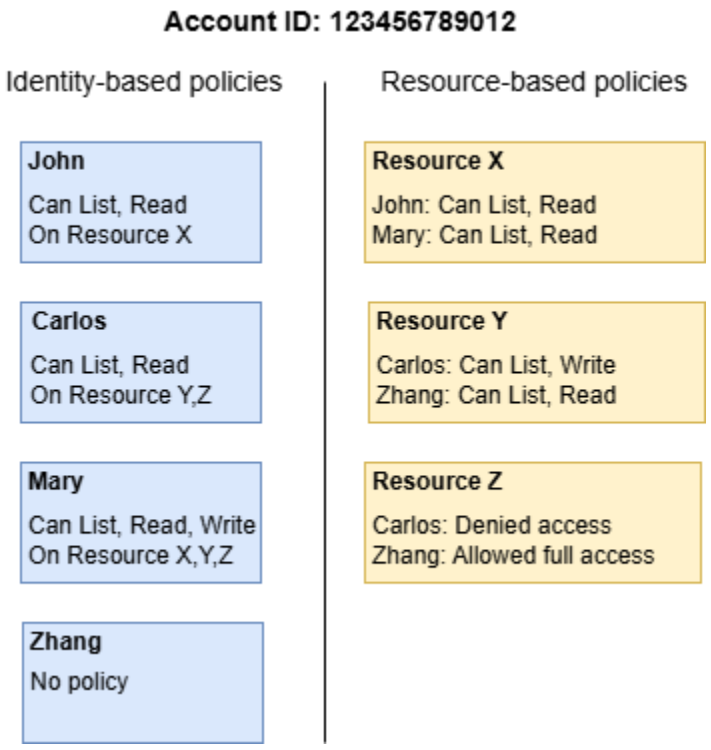
Avec les politiques basées sur une ressource, vous pouvez spécifier qui a accès à la ressource et quelles actions peuvent être exécutées. Pour savoir si les responsables de comptes situés en dehors de votre zone de confiance (organisation de confiance ou compte) ont accès à vos rôles, voir Qu'est-ce qu'IAM Access Analyzer ? . Resource-based les politiques sont intégrées uniquement, elles ne sont pas gérées.

> **Note**
>
> Resource-basedles politiques diffèrent des autorisations au niveau des ressources. Vous pouvez associer des politiques basées sur les ressources directement à une ressource, comme décrit dans cette rubrique. Resource-level les autorisations font référence à la possibilité d'utiliser des ARN pour spécifier des ressources individuelles dans une politique. Resource-based les politiques ne sont prises en charge que par certains AWS services. Pour obtenir la liste des services qui prennent en charge les politiques basées sur les ressources et les autorisations de niveau ressource, consultez AWS services qui fonctionnent avec IAM.

Pour découvrir comment les politiques basées sur l'identité et les politiques basées sur les ressources interagissent dans un même compte, veuillez consulter Évaluation des politiques pour les demandes au sein d'un même compte.

Pour découvrir comment les politiques interagissent entre comptes, veuillez consulter Cross-account logique d'évaluation des politiques.

Pour mieux comprendre ces concepts, voir la figure suivante. L'administrateur des `123456789012` politiques basées sur l'identité attachée au compte pour les utilisateurs `John`, `Carlos` et `Mary`. Certaines actions de ces politiques peuvent être effectuées sur des ressources spécifiques. Par exemple, l'utilisateur `John` peut effectuer certaines actions sur `Resource X`. Il s'agit d'une autorisation au niveau ressource dans une politique basée sur une identité. L'administrateur a également ajouté des politiques basées sur les ressources à `Resource X``Resource Y`, et. `Resource Z` Resource-based les politiques vous permettent de spécifier qui peut accéder à cette ressource. Par exemple, la politique basée sur les ressources sur `Resource X` accorde aux utilisateurs `John` et `Mary` l'accès en lecture à la ressource.



L'exemple de compte `123456789012` autorise les utilisateurs suivants à exécuter les actions répertoriées :

- John : John peut exécuter les actions d'affichage et de lecture sur `Resource X`. Cette autorisation lui est accordée par la politique basée sur l'identité sur son utilisateur et par la politique basée sur les ressources sur `Resource X`.
- Carlos : Carlos peut effectuer des actions d'affichage, de lecture et d'écriture sur `Resource Y`, mais se voit refuser l'accès à `Resource Z`. La politique basée sur l'identité sur Carlos lui permet d'effectuer les actions d'affichage et de lecture sur `Resource Y`. La politique basée sur la ressource `Resource Y` lui accorde également les autorisations d'écriture. Cependant, même si sa politique basée sur les identités lui permet l'accès à `Resource Z`, la politique basée sur les ressources `Resource Z` lui refuse l'accès. Un `Deny` explicite remplace une `Allow` et l'accès à `Resource Z` est refusé. Pour de plus amples informations, veuillez consulter Logique d'évaluation de politiques.
- Mary : Mary peut exécuter les opérations d'affichage, de lecture et d'écriture sur `Resource X`, `Resource Y` et `Resource Z`. Sa politique basée sur les identités lui permet plus d'actions sur plus de ressources que les politiques basées sur les ressources, mais aucune d'elles ne lui refuse l'accès.
- Zhang : Zhang dispose d'un accès complet à `Resource Z`. Zhang n'a pas de politiques basées sur l'identité, mais la politique basée sur les `Resource Z` ressources l'autorise un accès complet à la ressource. Zhang peut également effectuer les actions List (Liste) et Read (Lire) sur `Resource Y`.

Identity-based les politiques et les politiques basées sur les ressources sont toutes deux des politiques d'autorisation et sont évaluées conjointement. Pour une demande à laquelle seules les politiques d'autorisation s'appliquent, AWS vérifiez d'abord toutes les politiques pour un`Deny`. Le cas échéant, la demande est refusée. Ensuite, AWS vérifie chaque `Allow`. Si au moins une instruction de politique autorise l'action dans la demande, la demande est autorisée. Il n'est pas important que l'`Allow` soit dans la politique basée sur les identités ou dans celle basée sur les ressources.

> **Important**
>
> Cette logique s'applique uniquement lorsque la demande est effectuée au sein d'un même Compte AWS. Pour les demandes effectuées à partir d'un compte vers un autre, le demandeur `Account A` doit avoir une politique basée sur une identité qui leur permet d'adresser une demande à la ressource dans `Account B`. En outre, la politique basée sur les ressources dans `Account B` doivent autoriser le demandeur dans `Account A` à accéder à la ressource. Il doit y avoir des politiques dans les deux comptes qui autorisent l'opération, sinon la demande échoue. Pour plus d'informations sur l'utilisation des politiques basées sur

une ressource pour l'accès inter-compte, consultez Accès intercompte aux ressources dans IAM.

Un utilisateur qui dispose d'autorisations spécifiques peut demander une ressource à laquelle une politique d'autorisation est attachée. Dans ce cas, AWS évalue les deux ensembles d'autorisations pour déterminer s'il convient d'accorder l'accès à la ressource. Pour plus d'informations sur la manière dont les politiques sont évaluées, consultez Logique d'évaluation de politiques.

> **Note**
>
> Amazon S3 prend en charge les politiques basées sur l'identité et les politiques basées sur les ressources (appelées politiques de compartiment). En outre, Amazon S3 prend en charge un mécanisme d'autorisation appelé liste de contrôle d'accès (ACL) qui est indépendant des politiques et des autorisations IAM. Vous pouvez utiliser des politiques IAM combinées à des listes ACL (liste de contrôle d'accès) Amazon S3. Pour plus d'informations, veuillez consulter contrôle d'accès dans le guide de l'utilisateur du service de stockage simple Amazon.

Contrôlez l'accès aux AWS ressources à l'aide de politiques

Vous pouvez utiliser une politique pour contrôler l'accès aux ressources au sein d'IAM ou de la totalité d' AWS entre elles.

Pour utiliser une politique visant à contrôler l'accès AWS, vous devez comprendre comment AWS octroyer l'accès. AWS est composé de collections de ressources. Un utilisateur IAM est une ressource. Un compartiment Amazon S3 est une ressource. Lorsque vous utilisez l' AWS API, le AWS CLI, ou le Console de gestion AWS pour effectuer une opération (telle que la création d'un utilisateur), vous envoyez une demande pour cette opération. Votre demande spécifie une action, une ressource, une entité du principal (utilisateur ou rôle), un compte du principal et toutes les informations nécessaires relatives à la demande. Toutes ces informations fournissent le contexte.

AWS vérifie ensuite que vous (le principal) êtes authentifié (connecté) et autorisé (autorisé) à effectuer l'action spécifiée sur la ressource spécifiée. Lors de l'autorisation, AWS vérifie toutes les politiques qui s'appliquent au contexte de votre demande. La plupart des politiques sont stockées AWS sous forme de documents JSON et spécifient les autorisations pour les entités principales. Pour plus d'informations sur les types de politiques et les utilisations, veuillez consulter Politiques et autorisations dans Gestion des identités et des accès AWS.

AWS autorise la demande uniquement si chaque partie de votre demande est autorisée par les politiques. Pour afficher un schéma de ce processus, reportez-vous à Fonctionnement de IAM. Pour plus de détails sur AWS la manière de déterminer si une demande est autorisée, consultezLogique d'évaluation de politiques.

Lorsque vous créez une politique IAM, vous pouvez contrôler l'accès aux éléments suivants :

- Entités du principal – Contrôler ce que l'individu à l'origine de la requête (le principal) est autorisé à faire.
- Identités IAM : contrôlez à quelles identités IAM (groupes d'utilisateurs, utilisateurs et rôles) il est possible d'accéder et comment y accéder.
- IAM Policies (Politiques IAM) : contrôlez qui peut créer, modifier et supprimer les politiques gérées par les clients, et qui peut attacher et détacher toutes les politiques gérées.
- Ressources AWS : contrôlez qui a accès aux ressources à l'aide d'une politique basée sur les identités ou sur les ressources.
- AWS Accounts (Comptes) : contrôle si une requête est autorisée uniquement pour les membres d'un compte spécifique.

Les politiques vous permettent de spécifier qui a accès aux AWS ressources et quelles actions ils peuvent effectuer sur ces ressources. Chaque utilisateur IAM démarre sans autorisation. En d'autres termes, par défaut, les utilisateurs ne peuvent rien faire, pas même afficher leurs propres clés d'accès. Pour autoriser un utilisateur à effectuer une action, vous pouvez ajouter l'autorisation appropriée pour l'utilisateur (c'est-à-dire attacher une politique à celui-ci). Vous pouvez également ajouter l'utilisateur à un groupe d'utilisateurs disposant de l'autorisation prévue.

Par exemple, vous pourriez accorder à un utilisateur l'autorisation d'afficher ses propres clés d'accès. Vous pourriez également étendre cette autorisation et permettre également à chaque utilisateur de créer, mettre à jour et la supprimer ses propres clés.

Lorsque vous accordez des autorisations à un groupe d'utilisateurs, tous les utilisateurs de ce groupe obtiennent ces autorisations. Par exemple, vous pouvez autoriser le groupe d'utilisateurs Administrateurs à effectuer n'importe quelle action IAM sur n'importe quelle Compte AWS ressource. Un autre exemple : vous pouvez donner au groupe d'utilisateurs Managers (Gestionnaires) l'autorisation de décrire les instances Amazon EC2 de l' Compte AWS.

Pour de plus amples informations sur la façon de déléguer des autorisations de base à vos utilisateurs, groupes IAM et rôles, consultez Autorisations requises pour accéder aux autres

ressources IAM. Pour obtenir des exemples supplémentaires de politiques illustrant des autorisations de base, consultez Exemples de politiques de gestion de ressources IAM.

Contrôle de l'accès pour les entités du principal

Vous pouvez utiliser des politiques pour contrôler ce que la personne à l'origine de la demande (le principal) est autorisée à effectuer. Pour ce faire, vous devez attacher à l'identité de cette personne une politique basée sur l'identité (utilisateur, groupe d'utilisateurs ou rôle). Vous pouvez également utiliser une limite d'autorisations pour définir les autorisations maximales qu'une entité (utilisateur ou rôle) peut avoir.

Supposons, par exemple, que vous souhaitiez que l'utilisateur Zhang Wei ait un accès complet à Amazon DynamoDB CloudWatch, Amazon EC2 et Amazon S3. Vous pouvez créer deux politiques différentes pour pouvoir les diviser plus tard si vous avez besoin d'un ensemble d'autorisations pour un autre utilisateur. Vous pouvez également regrouper les deux autorisations dans une seule politique, puis associer cette dernière à l'utilisateur IAM nommé Zhang Wei. Vous pouvez aussi attacher une politique à un groupe d'utilisateurs auquel Zhang appartient, ou à un rôle que Zhang peut endosser. Ainsi, lorsque Zhang consulte le contenu d'un compartiment S3, ses demandes sont autorisées. S'il tente de créer un nouvel utilisateur IAM, sa demande est rejetée car il ne dispose pas de l'autorisation appropriée.

Vous pouvez utiliser une limite de permissions sur Zhang pour veiller à ce qu'il n'ait jamais accès au compartiment S3 `amzn-s3-demo-bucket1`. Pour ce faire, déterminez le nombre maximum d'autorisations dont vous souhaitez que Zhang dispose. Dans ce cas, vous pouvez contrôler ses actions à l'aide de ses politiques d'autorisations. Ici, vous veillez simplement à ce qu'il n'accède pas au compartiment confidentiel. Vous utilisez donc la politique suivante pour définir les limites de Zhang afin d'autoriser toutes les AWS actions pour Amazon S3 et quelques autres services, mais de refuser l'accès au compartiment `amzn-s3-demo-bucket1` S3. Étant donné que la limite d'autorisations ne permet pas tous les actions IAM, elle empêche Zhang de supprimer sa limite (ou celle d'une autre personne).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PermissionsBoundarySomeServices",
```

```
            "Effect": "Allow",
            "Action": [
                "cloudwatch:*",
                "dynamodb:*",
                "ec2:*",
                "s3:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "PermissionsBoundaryNoConfidentialBucket",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1",
                "arn:aws:s3:::amzn-s3-demo-bucket1/*"
            ]
        }
    ]
}
```

Lorsque vous attribuez une telle politique à un utilisateur en tant que limite d'autorisations, n'oubliez pas qu'elle n'accorde aucune autorisation. Elle définit le nombre maximum d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM. Pour plus d'informations sur les limites d'autorisations, consultez Limites d'autorisations pour les entités IAM.

Pour obtenir des informations détaillées sur les procédures mentionnées précédemment, reportez-vous aux ressources suivantes :

- Pour en savoir plus sur la création d'une politique IAM susceptible d'être attachée à un principal, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client.
- Pour apprendre à attacher une politique IAM à un principal, consultez Ajout et suppression d'autorisations basées sur l'identité IAM.
- Pour consulter un exemple de politique d'octroi d'un accès total à EC2, reportez-vous à Amazon EC2 : autorise l'accès EC2 complet dans une région spécifique, par programmation et dans la console.
- Pour autoriser un accès en lecture seule à un compartiment S3, utilisez les deux premières instructions de l'exemple de politique suivant : Amazon S3 : autorise l'accès en lecture et en écriture aux objets d'un compartiment S3, par programmation et dans la console.

- Pour consulter un exemple de politique permettant aux utilisateurs de définir leurs informations d'identification (leur mot de passe de console, leurs clés d'accès par programmation ou leurs dispositifs MFA, par exemple), veuillez consulter AWS : autorise les utilisateurs IAM authentifiés par MFA à gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

Contrôle de l'accès aux identités

Vous pouvez utiliser des politiques IAM pour contrôler ce que vos utilisateurs peuvent faire à une identité en créant une politique que vous attachez à tous les utilisateurs via un groupe d'utilisateurs. Pour cela, créez une politique qui limite les opérations susceptibles d'être effectuées sur une identité ou les autorisations d'accès.

Par exemple, vous pouvez créer un groupe d'utilisateurs nommé AllUsers, puis l'associer à tous les utilisateurs. Lorsque vous créez le groupe d'utilisateurs, vous pouvez donner à tous vos utilisateurs l'accès à la définition de leurs informations d'identification comme décrit dans la section précédente. Vous pouvez ensuite créer une politique qui interdit l'accès en modification du groupe d'utilisateurs, sauf si le nom d'utilisateur est inclus dans la condition de la politique. Mais cette partie de la politique interdit uniquement l'accès à tous, à l'exception des utilisateurs répertoriés. Vous devez également inclure les autorisations visant à permettre toutes les actions de gestion de groupe d'utilisateurs pour toutes les personnes du groupe. Enfin, vous attachez cette politique au groupe d'utilisateurs de façon à ce qu'elle soit appliquée à tous les utilisateurs. De cette façon, lorsqu'un utilisateur non spécifié dans la politique tente d'apporter des modifications au groupe d'utilisateurs, la demande est refusée.

Pour créer cette politique avec l'éditeur visuel

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de gauche, sélectionnez Policies (Politiques).

   Si vous sélectionnez Politiques pour la première fois, la page Bienvenue dans les politiques gérées s'affiche. Sélectionnez Get started (Mise en route).
3. Choisissez Create Policy (Créer une politique).
4. Dans la section Éditeur de politiques, choisissez l'option Visuel.
5. Dans Sélectionner un service, choisissez IAM.
6. Dans Actions autorisées, tapez **`group`** dans la zone de recherche. L'éditeur visuel affiche toutes les actions IAM qui contiennent le mot `group`. Sélectionnez toutes les cases à cocher.

7. Choisissez Ressources pour spécifier les ressources de votre politique. En fonction des actions que vous avez choisies, vous devriez voir les types de ressources groupe et utilisateur.

   - groupe — Choisissez Ajouter ARNs. Pour Ressource dans, sélectionnez l'option Tout compte. Cochez la case Tout nom de groupe avec chemin, puis saisissez le nom du groupe d’utilisateurs **`AllUsers`**. Choisissez ensuite Ajouter ARNs.
   - utilisateur : cochez la case en regard de N’importe quel/quelle dans ce compte.

   Une des actions que vous avez choisies, `ListGroups`, ne prend pas en charge l'utilisation de ressources spécifiques. Vous n'avez pas besoin de choisir Toutes les ressources pour cette action. Lorsque vous enregistrez votre politique ou que vous l'affichez dans l'éditeur JSON, vous pouvez voir que IAM crée automatiquement un nouveau bloc d'autorisation qui accorde à cette action une autorisation sur toutes les ressources.
8. Pour ajouter un autre bloc d'autorisation, choisissez Ajouter d'autres autorisations.
9. Choisissez Sélectionner un service, puis IAM.
10. Choisissez Actions autorisées, puis Basculer pour refuser les autorisations. Si vous procédez ainsi, l'ensemble du bloc est utilisé pour refuser des autorisations.
11. Dans la zone de recherche, saisissez **`group`**. L'éditeur visuel affiche toutes les actions IAM qui contiennent le mot `group`. Activez les cases à cocher en regard des actions suivantes :

    - CreateGroup
    - DeleteGroup
    - RemoveUserFromGroup
    - AttachGroupPolicy
    - DeleteGroupPolicy
    - DetachGroupPolicy
    - PutGroupPolicy
    - UpdateGroup
12. Choisissez Ressources pour spécifier les ressources de votre politique. En fonction des actions que vous avez choisies, vous devriez voir le type de ressource group. Choisissez Ajouter ARNs. Pour Ressource dans, sélectionnez l'option Tout compte. Pour Tout nom de groupe avec chemin, tapez le nom du groupe d'utilisateurs **`AllUsers`**. Choisissez ensuite Ajouter ARNs.
13. Choisissez Demander des conditions – facultatif, puis Ajouter une autre condition. Remplissez le formulaire avec les valeurs suivantes :

- Clé de condition : choisissez aws:username
- Qualifier (Qualificateur) : sélectionnez Default (Valeur par défaut)
- Opérateur — Choisissez StringNotEquals
- Valeur : tapez **`srodriguez`**, puis choisissez Ajouter pour ajouter une autre valeur. Tapez **`mjackson`**, puis choisissez Ajouter pour ajouter une autre valeur. Tapez **`adesai`**, puis choisissez Ajouter une condition.

Grâce à cette condition, l'accès est refusé aux actions spécifiées de gestion du groupe d'utilisateurs lorsque l'utilisateur à l'origine de l'action n'est pas inclus dans la liste. Dans la mesure où cette option refuse explicitement l'autorisation, elle remplace le bloc précédent qui a autorisé les utilisateurs à demander les actions. Les utilisateurs de la liste ne se voient pas refuser l'accès et ils sont autorisés dans le premier bloc d'autorisation, afin qu'ils puissent entièrement gérer le groupe d'utilisateurs.

14. Lorsque vous avez terminé, choisissez Next.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'option éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour de plus amples informations, veuillez consulter Restructuration de politique.

15. Sur la page Vérifier et créer, pour le Nom de la politique, tapez **`LimitAllUserGroupManagement`**. Pour la Description, saisissez **`Allows all users read-only access to a specific user group, and allows only specific users access to make changes to the user group`**. Vérifiez les Autorisations définies dans cette politique pour vous assurer que vous les avez accordées comme prévu. Ensuite, choisissez Créer une politique pour enregistrer votre nouvelle politique.
16. Attachez la politique à votre groupe d'utilisateurs. Pour plus d'informations, veuillez consulter Ajout et suppression d'autorisations basées sur l'identité IAM.

Vous pouvez aussi créer la même politique à l'aide de cet exemple de document de politique JSON. Pour afficher cette politique JSON, veuillez consulter IAM : autorise des utilisateurs spécifiques à gérer un groupe par programmation et dans la console. Pour obtenir des instructions détaillées sur la

création d'une politique à l'aide d'un document JSON, veuillez consulter the section called "Création de politiques à l'aide de l'éditeur JSON".

Contrôle de l'accès aux politiques

Vous pouvez contrôler la manière dont vos utilisateurs peuvent appliquer les politiques AWS gérées. Pour cela, attachez cette politique à tous vos utilisateurs. Idéalement, vous pouvez le faire en utilisant un groupe d'utilisateurs.

Par exemple, vous pouvez créer une politique qui permet aux utilisateurs d'associer uniquement IAMUserChangePasswordles politiques PowerUserAccess AWS gérées à un nouvel utilisateur, groupe d'utilisateurs ou rôle IAM.

Pour les politiques gérées par les clients, vous pouvez contrôler qui peut les créer, les mettre à jour et les supprimer. Vous pouvez contrôler qui peut attacher et détacher les politiques vers et depuis des entités de principal (groupes, utilisateurs et rôles IAM). Vous pouvez également contrôler quelles politiques peuvent être attachées ou détachées de ces entités.

Par exemple, vous pouvez autoriser un administrateur de compte à créer, mettre à jour et supprimer des politiques. Ensuite, vous accordez des autorisations à un responsable d'équipe ou un autre administrateur disposant de droits limités afin de lui permettre d'attacher et de détacher ces politiques des entités du principal gérées par l'administrateur doté de droits limités.

Pour de plus amples informations, veuillez consulter les ressources suivantes :

- Pour en savoir plus sur la création d'une politique IAM susceptible d'être attachée à un principal, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client.
- Pour apprendre à attacher une politique IAM à un principal, consultez Ajout et suppression d'autorisations basées sur l'identité IAM.
- Pour consulter un exemple de politique permettant de limiter l'utilisation des politiques gérées, reportez-vous à IAM : limite les politiques gérées pouvant être appliquées à un utilisateur, groupe ou rôle IAM.

Contrôle des autorisations pour la création, mise à jour et suppression de politiques gérées par le client

Vous pouvez utiliser des politiques IAM pour contrôler qui est autorisé à créer, mettre à jour et supprimer les politiques gérées par le client dans votre Compte AWS. La liste suivante répertorie les

opérations d'API ayant un lien direct avec la création, la mise à jour, la suppression et la gestion des politiques ou versions de politiques :

- CreatePolicy
- CreatePolicyVersion
- DeletePolicy
- DeletePolicyVersion
- SetDefaultPolicyVersion

Les opérations d'API ci-dessus correspondent à des actions que vous pouvez autoriser ou refuser autrement dit, des autorisations que vous pouvez accorder à l'aide d'une politique IAM.

Examinons l'exemple de politique suivant. Il autorise un utilisateur à créer, mettre à jour (autrement dit, créer une version de politique), supprimer et définir une version par défaut pour toutes les politiques gérées par le client dans l' Compte AWS. Cette politique permet également à l'utilisateur d'obtenir des politiques ou d'en afficher la liste. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

Example Exemple de politique qui autorise la création, mise à jour, suppression, obtention et définition de la version par défaut pour toutes les politiques

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:CreatePolicy",
      "iam:CreatePolicyVersion",
      "iam:DeletePolicy",
      "iam:DeletePolicyVersion",
      "iam:GetPolicy",
      "iam:GetPolicyVersion",
      "iam:ListPolicies",
      "iam:ListPolicyVersions",
      "iam:SetDefaultPolicyVersion"
```

```
    ],
    "Resource": "*"
  }
}
```

Vous pouvez créer des politiques qui limitent l'utilisation de ces opérations d'API aux seules politiques gérées que vous spécifiez. Par exemple, il est possible d'autoriser un utilisateur à définir la version par défaut et supprimer des versions de politique, mais uniquement pour certaines politiques gérées par le client. Pour ce faire, vous spécifiez l'ARN de la politique dans l'élément `Resource` de la politique qui accorde les autorisations.

L'exemple suivant illustre une politique qui autorise un utilisateur à supprimer des versions de politique et à définir la version par défaut. Mais ces actions sont uniquement autorisées pour les politiques gérées par le client comportant le chemin/TEAM-A/. L'ARN de la politique gérée par le client est spécifié dans l'élément `Resource` de la politique. Dans cet exemple, l'ARN inclut un chemin d'accès et un caractère générique et, par conséquent, correspond à toutes les politiques gérées par le client qui incluent le chemin /TEAM-A/. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

Pour de plus amples informations sur l'utilisation de chemins dans les noms de politiques gérées par le client, veuillez consulter Noms conviviaux et chemins.

Example Exemple de politique qui autorise la suppression de versions de politiques et la définition de la version par défaut pour des politiques spécifiques uniquement

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:DeletePolicyVersion",
            "iam:SetDefaultPolicyVersion"
        ],
        "Resource": "arn:aws:iam::111122223333:policy/TEAM-A/*"
    }
}
```

Contrôle des autorisations pour attacher et détacher des politiques gérées

Vous pouvez également utiliser des politiques IAM pour n'autoriser les utilisateurs qu'à utiliser des politiques gérées spécifiques. En effet, vous pouvez contrôler les autorisations qu'un utilisateur est autorisé à accorder à d'autres entités du principal.

La liste suivante répertorie les opérations d'API utilisées pour attacher et détacher des politiques gérées d'entités du principal :

- AttachGroupPolicy
- AttachRolePolicy
- AttachUserPolicy
- DetachGroupPolicy
- DetachRolePolicy
- DetachUserPolicy

Vous pouvez créer des politiques qui limitent l'utilisation de ces opérations d'API afin d'affecter uniquement les entités and/or principales des politiques gérées spécifiques que vous spécifiez. Par exemple, il est possible d'autoriser un utilisateur à attacher des politiques gérées, mais uniquement celles que vous spécifiez. Ou, il est possible d'autoriser un utilisateur à attacher des politiques gérées, mais uniquement aux entités du principal que vous spécifiez.

L'exemple suivant illustre une politique qui autorise un utilisateur à attacher des politiques gérées aux groupes et rôles IAM comportant le chemin /TEAM-A/ uniquement. Le groupe d'utilisateurs et le rôle ARNs sont spécifiés dans l'`Resource`élément de la politique. (Dans cet exemple, ils ARNs incluent un chemin et un caractère générique et correspondent ainsi à tous les groupes et rôles IAM qui incluent le chemin /TEAM-A/). Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

Example Exemple de politique qui autorise l'utilisateur à attacher des politiques gérées à des groupes d'utilisateurs ou rôles spécifiques uniquement

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
```

```
        "Effect": "Allow",
        "Action": [
            "iam:AttachGroupPolicy",
            "iam:AttachRolePolicy"
        ],
        "Resource": [
            "arn:aws:iam::111122223333:group/TEAM-A/*",
            "arn:aws:iam::111122223333:role/TEAM-A/*"
        ]
    }
}
```

Vous pouvez limiter encore davantage les actions de l'exemple précédent, de manière à n'affecter que des politiques spécifiques. En d'autres termes, vous pouvez contrôler les autorisations qu'un utilisateur est autorisé à attacher à d'autres entités du principal, en ajoutant une condition à la politique.

Dans l'exemple suivant, la condition garantit que les autorisations `AttachGroupPolicy` et `AttachRolePolicy` sont octroyées uniquement si la politique devant être attachée correspond à l'une des politiques spécifiées. La condition utilise la clé de condition `iam:PolicyARN` pour déterminer la ou les politiques qui peuvent être attachées. L'exemple de politique suivant développe l'exemple précédent. Il permet à un utilisateur de n'attacher que les politiques gérées qui comportent le chemin /TEAM-A/ aux seuls groupes et rôles IAM qui comportent le chemin /TEAM-A/. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachGroupPolicy",
            "iam:AttachRolePolicy"
        ],
        "Resource": [
            "arn:aws:iam::111122223333:group/TEAM-A/*",
            "arn:aws:iam::111122223333:role/TEAM-A/*"
        ],
```

```
        "Condition": {
            "ArnLike": {
                "iam:PolicyARN": "arn:aws:iam::111122223333:policy/TEAM-A/*"
            }
        }
    }
}
```

Cette politique utilise l'opérateur de condition `ArnLike`, mais vous pouvez également utiliser l'opérateur de condition `ArnEquals` car ces deux opérateurs de condition se comportent de manière identique. Pour de plus amples informations sur `ArnLike` et `ArnEquals`, veuillez consulter Opérateurs de condition d'Amazon Resource Name (ARN) dans la section Types de conditions des Références des éléments de politique.

Par exemple, vous pouvez limiter l'utilisation de ces actions aux politiques gérées que vous spécifiez. Pour ce faire, vous spécifiez l'ARN de la politique dans l'élément `Condition` de la politique qui accorde les autorisations. Par exemple, pour spécifier l'ARN d'une politique gérée par le client :

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::123456789012:policy/POLICY-NAME"}
}
```

Vous pouvez également spécifier l'ARN d'une stratégie AWS gérée dans l'`Condition`élément d'une stratégie. L'ARN d'une politique AWS gérée utilise l'alias spécial contenu `aws` dans l'ARN de la politique au lieu d'un identifiant de compte, comme dans cet exemple :

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::aws:policy/AmazonEC2FullAccess"}
}
```

Contrôle de l'accès aux ressources

Vous pouvez contrôler l'accès aux ressources à l'aide d'une politique basée sur les identités ou sur les ressources. Dans le cadre d'une politique basée sur les identités, vous attachez la politique à une identité et vous spécifiez à quelles ressources cette identité peut accéder. Dans le cadre d'une politique basée sur les ressources, vous attachez une politique à la ressource que vous souhaitez contrôler. Dans la politique, vous spécifiez quels principaux peuvent accéder à cette ressource. Pour de plus amples informations sur les deux types de stratégie, veuillez consulter Identity-based politiques et politiques basées sur les ressources.

Pour de plus amples informations, veuillez consulter les ressources suivantes :

- Pour en savoir plus sur la création d'une politique IAM susceptible d'être attachée à un principal, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client.
- Pour apprendre à attacher une politique IAM à un principal, consultez Ajout et suppression d'autorisations basées sur l'identité IAM.
- Amazon S3 prend en charge l'utilisation des politiques basées sur les ressources sur leurs compartiments. Pour de plus amples informations, veuillez consulter Exemples de politique de compartiment.

Des créateurs de ressource ne reçoivent pas automatiquement des autorisations

Si vous vous connectez à l'aide des Utilisateur racine d'un compte AWS informations d'identification, vous êtes autorisé à effectuer toute action sur les ressources appartenant au compte. Toutefois, ce n'est pas le cas pour les utilisateurs IAM. Un utilisateur IAM peut recevoir un accès pour créer une ressource, mais les autorisations de cet utilisateur, même pour cette ressource, se limitent à ce qui a été explicitement accordé. Cela signifie que si vous créez une ressource, par exemple un rôle IAM, vous n'avez pas automatiquement l'autorisation de modifier ou de supprimer ce rôle. De plus, votre autorisation peut être révoquée à tout moment par le propriétaire du compte ou par un autre utilisateur disposant d'un accès pour gérer vos autorisations.

Contrôle de l'accès aux principaux dans un compte spécifique

Vous pouvez accorder directement à des utilisateurs IAM de votre propre compte un accès à vos ressources. Si des utilisateurs d'un autre compte doivent accéder à vos ressources, vous pouvez créer un rôle IAM. Un rôle est une entité incluant des autorisations, mais qui n'est pas associée à un utilisateur spécifique. Les utilisateurs d'autres comptes peuvent ensuite endosser le rôle et accéder aux ressources en fonction des autorisations que vous avez attribuées à ce rôle. Pour plus d'informations, veuillez consulter Accès d'un utilisateur IAM à un autre Compte AWS que tu possèdes.

> **Note**
>
> Certains services prennent en charge les politiques basées sur les ressources, comme décrit dansIdentity-based politiques et politiques basées sur les ressources (Amazon S3, Amazon SNS et Amazon SQS, par ex.). Pour ces services, une alternative à l'utilisation de rôles consiste à attacher une politique à la ressource (compartiment, rubrique ou file d'attente) que

vous voulez partager. La politique basée sur les ressources peut spécifier le AWS compte autorisé à accéder à la ressource.

Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises

Utilisez les informations de la section suivante pour contrôler qui peut accéder à vos utilisateurs et rôles IAM, ainsi que les ressources auxquelles vos utilisateurs et rôles peuvent accéder. Pour des informations plus générales et des exemples de contrôle de l'accès à d'autres AWS ressources, y compris à d'autres ressources IAM, consultezTags pour Gestion des identités et des accès AWS resources.

> **Note**
>
> Pour plus d'informations sur la sensibilité à la casse des clés de balises et les valeurs de clés de balises, consultez Case sensitivity (français non garanti).

Les balises peuvent être attachées à la ressource IAM, transmises dans la demande ou attachées au principal qui effectue la demande. Un utilisateur ou rôle IAM peut être à la fois une ressource et un principal. Par exemple, vous pouvez écrire une politique qui permet à un utilisateur de répertorier les groupes d'un utilisateur. Cette opération est autorisée uniquement si l'utilisateur effectuant la demande (le principal) a la même balise `project=blue` que l'utilisateur qu'il essaie de consulter. Dans cet exemple, l'utilisateur peut afficher l'appartenance au groupe de n'importe quel utilisateur, y compris lui-même, tant qu'il travaille sur le même projet.

Pour contrôler l'accès basé sur des balises, vous devez fournir les informations des balises dans l'élément de condition d'une politique. Lorsque vous créez une politique IAM, vous pouvez utiliser des balises IAM et la clé de condition de balise associée pour contrôler l'accès à chacun des éléments suivants :

- Resource (Ressource) : contrôlez l'accès aux ressources d'utilisateur ou de rôle en fonction de leurs balises. Pour ce faire, utilisez la clé de *`key-name`* condition aws :ResourceTag/pour spécifier la paire clé-valeur de balise qui doit être attachée à la ressource. Pour de plus amples informations, veuillez consulter Contrôle de l'accès aux AWS ressources.
- Request (Demande) : contrôlez les balises pouvant être transmises dans une demande IAM. Pour ce faire, utilisez la clé de *`key-name`* condition aws :RequestTag/pour spécifier les balises qui

peuvent être ajoutées, modifiées ou supprimées pour un utilisateur ou un rôle IAM. Cette clé est utilisée de la même manière pour les ressources IAM et les autres AWS ressources. Pour de plus amples informations, veuillez consulter Contrôle de l'accès lors AWS des demandes.

- Principal : contrôlez ce que la personne à l'origine de la demande (le principal) est autorisée à faire en fonction des balises qui sont attachées à l'utilisateur ou au rôle IAM de cette personne. Pour ce faire, utilisez la clé de `key-name` condition aws :PrincipalTag/pour spécifier les balises qui doivent être attachées à l'utilisateur ou au rôle IAM avant que la demande ne soit autorisée.
- N'importe quelle partie du processus d'autorisation : utilisez la clé de TagKeys condition aws : pour contrôler si des clés de balise spécifiques peuvent être utilisées dans une demande ou par un principal. Dans ce cas, la valeur de la clé n'importe pas. Cette clé se comporte de la même manière pour IAM et les autres AWS services. Toutefois, lorsque vous balisez un utilisateur dans IAM, cette action contrôle également si le principal peut adresser la demande à n'importe quel service. Pour plus d'informations, veuillez consulter Contrôle de l'accès en fonction des clés de balise.

Vous pouvez créer une politique IAM à l'aide de l'éditeur visuel, à l'aide de JSON ou en important une politique gérée existante. Pour en savoir plus, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client.

> **Note**
>
> Vous pouvez également transmettre des balises de session lorsque vous endossez un rôle IAM ou fédérez un utilisateur. Celles-ci ne sont valides que pendant la durée de la session.

Contrôle de l'accès pour les principaux IAM

Vous pouvez contrôler ce que le principal est autorisé à faire en fonction des balises attachées à l'identité de cette personne.

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui permet à tout utilisateur de ce compte d'afficher l'appartenance au groupe de n'importe quel utilisateur, y compris lui-même, tant qu'il travaille sur le même projet. Cette opération est autorisée uniquement lorsque la balise de ressource de l'utilisateur et la balise du principal ont la même valeur pour la clé de balise `project`. Pour utiliser cette politique, remplacez celle de `italicized placeholder text` l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "iam:ListGroupsForUser",
            "Resource": "arn:aws:iam::111222333444:user/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/project":
 "${aws:PrincipalTag/project}"}
            }
        }]
}
```

Contrôle de l'accès en fonction des clés de balise

Vous pouvez utiliser les balises dans vos politiques IAM pour contrôler si des clés de balise spécifiques peuvent être utilisées sur une ressource, dans une requête ou par un principal.

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise la suppression de la balise avec la clé `temporary` uniquement, des utilisateurs. Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:UntagUser",
        "Resource": "*",
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys":
 ["temporary"]}}
    }]
```

```
}
```

Contrôle de l'accès aux AWS ressources à l'aide de balises

Vous pouvez utiliser des balises pour contrôler l'accès à vos AWS ressources qui prennent en charge le balisage, y compris les ressources IAM. Vous pouvez baliser des utilisateurs et des rôles IAM pour contrôler les éléments auxquels ils peuvent accéder. Pour savoir comment baliser des utilisateurs et des rôles IAM, consultez Tags pour Gestion des identités et des accès AWS resources. En outre, vous pouvez contrôler l'accès aux ressources IAM suivantes : politiques gérées par le client, fournisseurs d'identité IAM, profils d'instance, certificats de serveur et dispositifs MFA virtuels. Pour accéder à un didacticiel sur la création et le test d'une politique permettant aux rôles IAM avec des balises principaux d'accéder aux ressources avec des balises correspondantes, veuillez consulter Tutoriel IAM : définir les autorisations d'accès AWS ressources basées sur des balises. Utilisez les informations de la section suivante pour contrôler l'accès à d'autres AWS ressources, y compris les ressources IAM, sans baliser les utilisateurs ou les rôles IAM.

Avant d'utiliser des balises pour contrôler l'accès à vos AWS ressources, vous devez comprendre comment AWS octroyer l'accès. AWS est composé de collections de ressources. Une instance Amazon EC2 est une ressource. Un compartiment Amazon S3 est une ressource. Vous pouvez utiliser l' AWS API AWS CLI, le ou le Console de gestion AWS pour effectuer une opération, telle que la création d'un compartiment dans Amazon S3. Dans ce cas, vous envoyez une demande pour cette opération. Votre demande spécifie une action, une ressource, une entité mandataire (utilisateur ou rôle), un compte principal et toutes les informations nécessaires relatives à la demande. Toutes ces informations fournissent le contexte.

AWS vérifie ensuite que vous (l'entité principale) êtes authentifié (connecté) et autorisé (autorisé) à effectuer l'action spécifiée sur la ressource spécifiée. Lors de l'autorisation, AWS vérifie toutes les politiques qui s'appliquent au contexte de votre demande. La plupart des politiques sont stockées AWS sous forme de documents JSON et spécifient les autorisations pour les entités principales. Pour plus d'informations sur les types de politiques et les utilisations, veuillez consulter Politiques et autorisations dans Gestion des identités et des accès AWS.

AWS autorise la demande uniquement si chaque partie de votre demande est autorisée par les politiques. Pour afficher un schéma et en savoir plus sur l'infrastructure IAM, consultez Fonctionnement de IAM. Pour obtenir des détails sur la façon dont IAM détermine si une demande est autorisée, reportez-vous à Logique d'évaluation de politiques.

Les balises sont une autre considération concernant ce processus, car elles peuvent être attachées à la ressource ou transmises dans la demande aux services qui prennent en charge le balisage. Pour contrôler l'accès basé sur des balises, vous devez fournir les informations des balises dans l'élément de condition d'une politique. Pour savoir si un AWS service prend en charge le contrôle d'accès à l'aide de balises, consultez AWS services qui fonctionnent avec IAM et recherchez les services dont la valeur est « Oui » dans la colonne ABAC. Choisissez le nom du service pour afficher la documentation sur l'autorisation et le contrôle d'accès de ce service.

Vous pouvez ensuite créer une politique IAM qui autorise ou refuse l'accès à une ressource en fonction de la balise de cette ressource. Dans cette politique , vous pouvez utiliser des clés de condition de balise pour contrôler l'accès à l'un des éléments suivants :

- Ressource : contrôlez l'accès aux ressources de AWS service en fonction des balises figurant sur ces ressources. Pour ce faire, utilisez la clé de *key-name* condition aws :ResourceTag/pour déterminer s'il convient d'autoriser l'accès à la ressource en fonction des balises associées à la ressource.
- Request (Demande) : contrôle les balises qui peuvent être transmises dans une demande. Pour ce faire, utilisez la clé de *key-name* condition aws :RequestTag/pour spécifier les paires clé-valeur de balise qui peuvent être transmises dans une demande de balisage d'une AWS ressource.
- N'importe quelle partie du processus d'autorisation : utilisez la clé de TagKeys condition aws : pour contrôler si des clés de balise spécifiques peuvent figurer dans une demande.

Vous pouvez créer une politique IAM visuellement, à l'aide de JSON, ou en important une politique gérée existante. Pour en savoir plus, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client.

> **Note**
>
> Certains services permettent aux utilisateurs d'attribuer des balises au moment de la création de la ressource (s'ils ont les autorisations d'utiliser l'action qui crée la ressource).

Contrôle de l'accès aux AWS ressources

Vous pouvez utiliser des conditions dans vos politiques IAM pour contrôler l'accès aux AWS ressources en fonction des balises associées à ces ressources. Pour ce faire, vous pouvez utiliser la clé de condition globale `aws:ResourceTag/`*`tag-key`* ou une clé spécifique au service. Certains

services prennent en charge uniquement la version spécifique au service de cette clé et non la version globale.

> **Warning**
>
> N'essayez pas de contrôler les personnes susceptibles de faire passer un rôle en étiquetant ce rôle, puis en utilisant la clé de condition `ResourceTag` dans une politique avec l'action `iam:PassRole`. Cette approche ne produit pas des résultats fiables. Pour de plus amples informations sur les autorisations requises pour transmettre un rôle à un service, veuillez consulter Accorder à un utilisateur l'autorisation de transmettre un rôle à un AWS web.

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise le démarrage ou l'arrêt d'instances Amazon EC2. Ces opérations sont autorisées uniquement si la balise d'instance `Owner` a la valeur du nom d'utilisateur. Cette politique définit des autorisations pour l'accès à la console et par programmation.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/Owner": "${aws:username}"}
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

Vous pouvez rattacher cette politique aux utilisateurs IAM de votre compte. Si un utilisateur nommé `richard` tente de démarrer une instance Amazon EC2, l'instance doit être balisée `0wner=richard` ou `owner=richard`. Dans le cas contraire, il se verra refuser l'accès. La clé de balise `0wner` correspond à la fois à `0wner` et `owner`, car les noms de clé de condition ne sont pas sensibles à la casse. Pour de plus amples informations, veuillez consulter Éléments de la politique JSON IAM : Condition.

Cet exemple montre comment créer une politique basée sur l'identité qui utilise la balise de principal `team` dans l'ARN de la ressource. Cette politique autorise la suppression des files d'attente Amazon Simple Queue Service, mais uniquement si le nom de la file d'attente commence par le nom de l'équipe suivi de `-queue`. Par exemple, `qa-queue` si `qa` est le nom de l'équipe pour la balise de principal `team`.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Sid": "AllQueueActions",
        "Effect": "Allow",
        "Action": "sqs:DeleteQueue",
        "Resource": "arn:aws:sqs:us-east-2:111122223333:${aws:PrincipalTag/team}-
queue"
      }
}
```

Contrôle de l'accès lors AWS des demandes

Vous pouvez utiliser des conditions dans vos politiques IAM pour contrôler les paires clé-valeur de balise qui peuvent être transmises dans une demande qui applique des balises à une ressource. AWS

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'utilisation de l'action `CreateTags` d'Amazon EC2 pour attacher des balises à une instance. Vous pouvez attacher des balises uniquement si la balise contient la clé `environment` et les valeurs `production` ou `preprod`. En tant que bonne pratique, utilisez le modificateur `ForAllValues` avec la clé de condition `aws:TagKeys` pour indiquer que seule la clé `environment` est autorisée

dans la demande. Cela empêche les utilisateurs d'inclure d'autres clés, notamment en utilisant accidentellement `Environment` au lieu de `environment`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "ec2:CreateTags",
        "Resource": "arn:aws:ec2:*:*:instance/*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/environment": [
                    "preprod",
                    "production"
                ]
            },
            "ForAllValues:StringEquals": {"aws:TagKeys": "environment"}
        }
    }
}
```

Contrôle de l'accès en fonction des clés de balise

Vous pouvez utiliser une condition dans vos politiques IAM pour contrôler si des clés de balise spécifiques peuvent être utilisées dans une requête.

Lorsque vous utilisez des politiques pour contrôler l'accès à l'aide de balises, nous vous recommandons d'utiliser la clé de `aws:TagKeys` condition. AWS les services qui prennent en charge les balises peuvent vous permettre de créer plusieurs noms de clés de balise qui ne diffèrent qu'au cas par cas, par exemple en balisant une instance `stack=production` Amazon EC2 avec et. `Stack=test` Les noms de clé ne sont pas sensibles à la casse dans les conditions des politiques. Cela signifie que si vous spécifiez `"aws:ResourceTag/TagKey1": "Value1"` dans l'élément de condition de votre politique, la condition correspond à une clé de balise de ressource nommée `TagKey1` ou `tagkey1`, mais pas aux deux. Pour éviter la duplication de balises avec une clé qui ne varie que par la casse, utilisez la condition `aws:TagKeys` pour définir les clés de balise que vos utilisateurs peuvent appliquer, ou utilisez les politiques de balises, disponibles

avec AWS Organizations. Pour en savoir plus, consultez Stratégies de balisage dans le Guide de l'utilisateur AWS Organizations .

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise la création et la balise d'un secret Secrets Manager, mais uniquement avec les clés de balise `environment` ou `cost-center`. La condition `Null` garantit que la condition a la valeur `false` s'il n'y a pas de balises dans la demande.

```
{
        "Effect": "Allow",
        "Action": [
            "secretsmanager:CreateSecret",
            "secretsmanager:TagResource"
        ],
        "Resource": "*",
        "Condition": {
            "Null": {
                "aws:TagKeys": "false"
            },
            "ForAllValues:StringEquals": {
                "aws:TagKeys": [
                    "environment",
                    "cost-center"
                ]
            }
        }
}
```

Accès intercompte aux ressources dans IAM

Pour certains AWS services, vous pouvez accorder un accès intercomptes à vos ressources à l'aide d'IAM. Pour cela, vous pouvez attacher une politique de ressources directement à la ressource que vous voulez partager, ou bien utiliser un rôle en tant que proxy.

Pour partager directement une ressource, la ressource que vous voulez partager doit prendre en charge les politiques basées sur les ressources. Contrairement à une politique basée sur l'identité pour un rôle, une politique basée sur une ressource spécifie qui (quel principal) peut accéder à cette ressource.

Utilisez un rôle en tant que proxy lorsque vous voulez accéder aux ressources d'un autre compte qui ne prend pas en charge les politiques basées sur les ressources.

Pour en savoir plus sur les différences entre ces types de politiques, consultez Identity-based politiques et politiques basées sur les ressources.

> **Note**
>
> Les politiques IAM basées sur les ressources et les rôles ne délèguent l'accès entre les comptes qu'au sein d'une seule partition. Par exemple, vous avez un compte dans la région USA Ouest (Californie du Nord) dans la partition `aws` standard. Vous avez également un compte dans la région Chine dans la partition `aws-cn`. Vous ne pouvez pas appliquer une politique basée sur les ressources à votre compte en Chine pour autoriser les utilisateurs à accéder à votre compte standard AWS .

Cross-account accès à l'aide de rôles

Les AWS services ne prennent pas tous en charge les politiques basées sur les ressources. Pour ces services, vous pouvez utiliser des rôles IAM intercomptes afin de centraliser la gestion des autorisations lorsque vous fournissez un accès intercompte à plusieurs services. Un rôle IAM intercomptes est un rôle IAM qui inclut une politique de confiance qui permet aux responsables IAM d'un autre AWS compte d'assumer ce rôle. En termes simples, vous pouvez créer un rôle dans un AWS compte qui délègue des autorisations spécifiques à un autre AWS compte.

Pour de plus amples informations sur l'attachement d'une politique à une identité IAM, veuillez consulter Gestion des politiques IAM.

> **Note**
>
> Lorsqu'un principal passe à un rôle pour utiliser temporairement les autorisations de ce rôle, il abandonne ses autorisations d'origine et reprend les autorisations attribuées au rôle qu'il a assumé.

Examinons le processus global tel qu'il s'applique au logiciel du partenaire APN qui doit accéder à un compte client.

1. Le client crée un rôle IAM dans son propre compte avec une politique qui autorise l'accès aux ressources Amazon S3 dont le partenaire APN a besoin. Dans cet exemple, le nom du rôle est `APNPartner`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name"
            ]
        }
    ]
}
```

2. Le client précise ensuite que le rôle peut être assumé par le AWS compte du partenaire en fournissant l' Compte AWS identifiant du partenaire APN dans la politique de confiance du APN`Partner` rôle.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/APN-user-name"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

3. Le client donne l'Amazon Resource Name (ARN) du rôle au partenaire APN. L'ARN est le nom entièrement qualifié du rôle.

```
arn:aws:iam::Customer-Account-ID:role/APNPartner
```

> **Note**
>
> Nous vous recommandons d'utiliser un ID externe dans les situations à locataires multiples. Pour en savoir plus, consultez Accès à Comptes AWS appartenant à des tiers.

4. Lorsque le logiciel du partenaire APN doit accéder au compte du client, le logiciel appelle l'AssumeRoleAPI AWS Security Token Service avec l'ARN du rôle dans le compte du client. STS renvoie un AWS identifiant temporaire qui permet au logiciel de faire son travail.

Pour un autre exemple de l'octroi d'un accès intercompte à l'aide de rôles, consultez Accès d'un utilisateur IAM à un autre Compte AWS que tu possèdes. Vous pouvez également suivre le tutoriel Tutoriel IAM : accès délégué via AWS comptes utilisant des rôles IAM.

Cross-account accès à l'aide de politiques basées sur les ressources

Lorsqu'un compte accède à une ressource par le biais d'un autre compte à l'aide d'une politique basée sur les ressources, le principal continue d'utiliser le compte approuvé sans devoir renoncer à ses autorisations au profit des autorisations du rôle. Autrement dit, le principal continue d'avoir accès aux ressources du compte approuvé tout en ayant accès à la ressource du compte d'approbation. Cela est utile pour les tâches de copie d'informations de ou vers la ressources partagée de l'autre compte.

Les principaux que vous pouvez spécifier dans une politique basée sur les ressources incluent les comptes, les utilisateurs IAM, les principaux utilisateurs AWS STS fédérés, les principaux fédérés SAML, les principaux fédérés OIDC, les rôles IAM, les sessions avec rôles assumés ou les services. AWS Pour plus d'informations, consultez Spécification d'un principal.

Pour savoir si les principaux des comptes situés en dehors de votre zone d'approbation (organisation ou compte d'approbation) peuvent accéder à vos rôles et les assumer, consultez Identification des ressources partagées avec une entité externe.

La liste suivante inclut certains des AWS services qui prennent en charge les politiques basées sur les ressources. Pour obtenir la liste complète du nombre croissant de AWS services qui prennent en charge l'attachement de politiques d'autorisation aux ressources plutôt qu'aux principaux, consultez AWS services qui fonctionnent avec IAM et recherchez les services dont la valeur est Oui dans la colonne Basée sur les ressources.

- Compartiments Amazon S3 : la politique est attachée au compartiment, mais la politique contrôle l'accès au compartiment et aux objets qu'il contient. Pour plus d'informations, consultez Politiques de compartiment pour Amazon S3 dans le Guide de l'utilisateur Amazon Simple Storage Service. Dans certains cas, il peut être préférable d'utiliser des rôles pour l'accès entre comptes à Amazon S3. Pour plus d'informations, veuillez consulter les exemples de démonstration dans le guide de l'utilisateur du service de stockage simple Amazon.
- Rubriques relatives à Amazon Simple Notification Service (Amazon SNS) — Pour plus d'informations, consultez la section Exemples de cas relatifs au contrôle d'accès Amazon SNS dans le manuel Amazon Simple Notification Service Developer Guide.
- Files d'attente Amazon Simple Queue Service (Amazon SQS) — Pour plus d'informations, consultez l'Annexe : Le langage de politique d'accès du guide du développeur Amazon Simple Queue Service.

Resource-based politiques de délégation AWS des autorisations

Si une ressource accorde des autorisations aux principaux de votre compte, vous pouvez alors déléguer ces autorisations à des identités IAM spécifiques. Les identités sont des utilisateurs, des groupes d'utilisateurs ou des rôles de votre compte. Vous déléguez des autorisations en attachant une politique à l'identité. Vous pouvez accorder autant d'autorisations que le compte propriétaire de la ressource le permet.

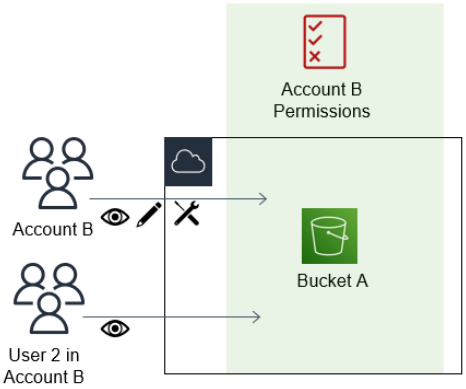
> **Important**
>
> En cas d'accès intercompte, un principal a besoin d'une `Allow` dans la politique d'identité intégrée et d'une politique basée sur les ressources.

Supposons qu'une politique basée sur une ressource accorde à tous les principaux de votre compte un accès administratif complet à une ressource. Vous pouvez ensuite déléguer l'accès complet, l'accès en lecture seule ou tout autre accès partiel aux principaux utilisateurs de votre compte. AWS Sinon, si la politique basée sur la ressource autorise uniquement les autorisations d'affichage de liste, vous pouvez déléguer uniquement ce type d'accès. Si vous essayez de déléguer davantage d'autorisations que n'en possède votre compte, l'accès de vos principaux restera limité à l'affichage de liste.

Pour plus d'informations sur la façon dont ces décisions sont prises, consultez Identification d'une demande autorisée ou refusée dans un compte.

> **Note**
>
> Les politiques IAM basées sur les ressources et les rôles ne délèguent l'accès entre les comptes qu'au sein d'une seule partition. Par exemple, vous ne pouvez pas ajouter d'accès entre comptes entre un compte dans la partition `aws` standard et un compte dans la partition `aws-cn`.

Par exemple, supposons que vous gériez `AccountA` et `AccountB`. Dans AccountA, vous avez un compartiment Amazon S3 nommé `BucketA`.



1. Vous attachez une politique basée sur les ressources à `BucketA` qui permet à tous les principaux de AccountB d'avoir un accès complet aux objets de votre compartiment. Ils peuvent créer, lire ou supprimer les objets de ce compartiment.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalAccess",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::BucketA/*"
        }
    ]
```

```
}
```

AccountA accorde à AccountB un accès complet à BucketA en désignant AccountB comme principal dans la politique basée sur les ressources. Par conséquent, AccountB est autorisé à effectuer toute action sur BucketA, et l'administrateur de AccountB peut déléguer l'accès à ses utilisateurs dans AccountB.

L'utilisateur root de AccountB dispose de toutes les autorisations accordées au compte. Par conséquent, l'utilisateur root a un accès complet à BucketA.

2. Dans AccountB, attachez une politique à l'utilisateur IAM nommé User2. Cette politique permet à l'utilisateur d'accéder en lecture seule aux objets de BucketA. Cela signifie que User2 peut afficher les objets, mais ne peut pas les créer, les modifier ou les supprimer.

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": [
           {
               "Effect" : "Allow",
               "Action" : [
                   "s3:Get*",
                   "s3:List*" ],
                   "Resource" : "arn:aws:s3:::BucketA/*"
           }
       ]
   }
   ```

   Le niveau d'accès maximal que AccountB peut déléguer est le niveau d'accès accordé au compte. Dans ce cas, la politique basée sur les ressources a accordé un accès complet à AccountB, mais User2 ne dispose que d'un accès en lecture seule.

   L'administrateur de AccountB n'accorde pas l'accès à User1. Par défaut, les utilisateurs n'ont pas d'autorisations, sauf celles qui sont accordées explicitement, et ainsi User1 n'a pas accès à BucketA.

IAM évalue les autorisations d'un principal au moment où il fait une demande. Si vous utilisez des caractères génériques (*) pour donner aux utilisateurs un accès complet à vos ressources, les administrateurs peuvent accéder à toutes les ressources auxquelles votre AWS compte a accès. Cela est vrai même pour les ressources que vous ajoutez ou auxquelles vous accédez après la création de la politique de l'utilisateur.

Dans l'exemple précédent, si AccountB avait attaché à User2 une politique permettant un accès complet à toutes les ressources de tous les comptes, User2 aurait automatiquement eu accès à toutes les ressources auxquelles AccountB a accès. Cela inclut l'accès à BucketA et l'accès à toutes les autres ressources accordé par les politiques basées sur les ressources dans AccountA.

Pour plus d'informations sur l'utilisation des rôles de manière plus complexe, comme accorder l'accès à des applications et des services, consultez Scénarios courants de rôles IAM.

> **Important**
>
> N'accordez l'accès qu'aux entités auxquelles vous faites confiance et accordez l'accès minimal nécessaire. Chaque fois que l'entité de confiance est un autre AWS compte, n'importe quel responsable IAM peut avoir accès à votre ressource. Le AWS compte sécurisé ne peut déléguer l'accès que dans la mesure où l'accès lui a été accordé ; il ne peut pas déléguer plus d'accès que le compte lui-même.

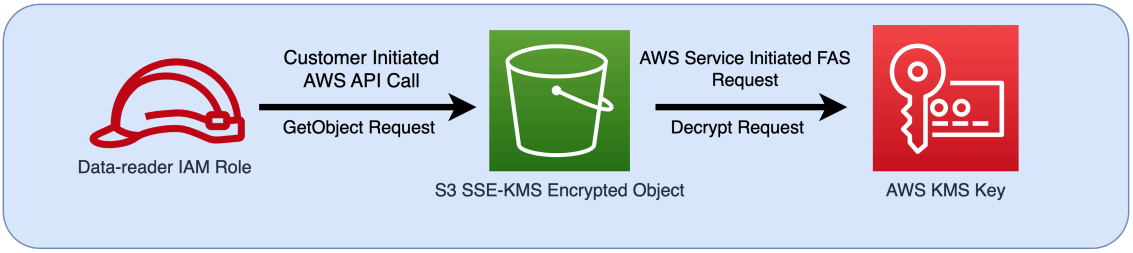
Pour plus d'informations sur les autorisations, les politiques et le langage de politique d'autorisation que vous utilisez pour écrire des politiques, consultez Gestion des accès pour AWS resources.

Transmission des sessions d'accès

Les sessions d'accès direct (FAS) sont une technologie IAM utilisée par les AWS services pour transmettre votre identité, vos autorisations et vos attributs de session lorsqu'un AWS service fait une demande en votre nom. FAS utilise les autorisations de l'identité appelant un AWS service, combinées à l'identité d'un AWS service pour adresser des demandes aux services en aval. Les demandes FAS ne sont adressées aux AWS services pour le compte d'un responsable IAM qu'après qu'un service a reçu une demande nécessitant des interactions avec d'autres AWS services ou ressources pour être traitée. Lorsqu'une demande de FAS est effectuée :

- Le service qui reçoit la demande initiale d'un principal IAM vérifie les autorisations de ce dernier.
- Le service qui reçoit une demande de FAS ultérieure vérifie également les autorisations du même principal IAM.

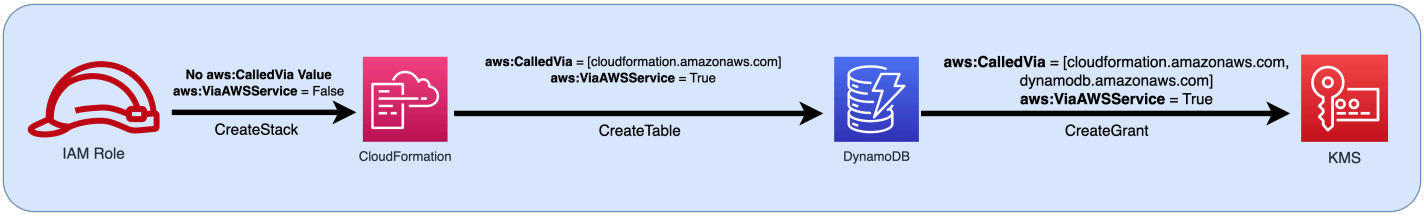
Par exemple, le FAS est utilisé par Amazon S3 pour passer des appels AWS Key Management Service afin de déchiffrer un objet alors qu'il SSE-KMS était utilisé pour le chiffrer. Lors du téléchargement d'un objet SSE-KMS chiffré, un rôle nommé data-reader fait GetObject appel à l'objet par rapport à Amazon S3, mais pas AWS KMS directement. Après avoir reçu la GetObject demande et autorisé le lecteur de données, Amazon S3 envoie une demande FAS AWS KMS à afin de déchiffrer l'objet Amazon S3. Lorsque KMS reçoit la demande de FAS, il vérifie les autorisations du rôle et n'autorise la demande de déchiffrement que si le lecteur de données dispose des autorisations correctes sur la clé KMS. Les demandes adressées à Amazon S3 et à Amazon S3 AWS KMS sont autorisées à l'aide des autorisations du rôle et ne sont acceptées que si le lecteur de données possède des autorisations à la fois sur l'objet Amazon S3 et sur la clé AWS KMS.



> **Note**
>
> Des demandes de FAS supplémentaires peuvent être effectuées par les services qui ont reçu une demande de FAS. Dans ce cas, le principal demandeur doit disposer d'autorisations pour tous les services appelés par la FAS.

Conditions des demandes de FAS et de la politique IAM

Lorsque des demandes de FAS sont effectuées, les clés de condition aws : CalledVia, aws : CalledViaFirst, et aws : CalledViaLast sont renseignées avec le principal du service qui a initié l'appel FAS. La valeur de la clé de condition aws:ViaAWSService est définie sur `true` chaque fois qu'une demande de FAS est effectuée. Dans le schéma suivant, aucune clé de `aws:ViaAWSService` condition `aws:CalledVia` ou aucune clé de condition n'est définie pour la requête CloudFormation directe. Lorsque CloudFormation DynamoDB effectue des requêtes FAS en aval pour le compte du rôle, les valeurs de ces clés de condition sont renseignées.



Pour permettre à une demande de FAS d'être effectuée alors qu'elle serait autrement refusée par une déclaration de politique de refus avec une clé de condition testant les adresses IP source ou les VPC source, vous devez utiliser des clés de condition pour fournir une exception pour les demandes de FAS dans votre politique de refus. Cela peut être fait pour toutes les demandes de FAS en utilisant la clé de condition `aws:ViaAWSService`. Pour autoriser uniquement des AWS services spécifiques à effectuer des demandes FAS, utilisez`aws:CalledVia`.

> **Important**
>
> Lorsqu'une demande de FAS est effectuée après qu'une demande initiale a été effectuée via un point de terminaison d'un VPC, les valeurs des clés de condition pour `aws : SourceVpce`, `aws : SourceVpc` et `aws : VpcSourceIp` de la demande initiale ne sont pas utilisées dans les demandes de FAS. Lorsque vous rédigez des politiques utilisant `aws:VpcSourceIp` ou `aws:SourceVpce` pour accorder un accès conditionnel, vous devez également utiliser `aws:ViaAWSService` ou `aws:CalledVia` pour autoriser les demandes de FAS. Lorsqu'une demande FAS est faite après la réception d'une demande initiale par un point de terminaison du AWS service public, les demandes FAS suivantes seront effectuées avec la même valeur de clé de `aws:SourceIP` condition.

Exemple : autoriser l'accès à Amazon S3 depuis un VPC ou avec la FAS

Dans l'exemple de politique IAM suivant, les demandes Amazon S3 GetObject et Athena ne sont autorisées que si elles proviennent de points de terminaison VPC attachés à*`example_vpc`*, ou s'il s'agit d'une demande FAS faite par Athena.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Sid": "OnlyAllowMyIPs",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*",
        "athena:StartQueryExecution",
        "athena:GetQueryResults",
        "athena:GetWorkGroup",
        "athena:StopQueryExecution",
        "athena:GetQueryExecution"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceVPC": [
          "vpc-111bbb22"
          ]
        }
      }
    },
    {
      "Sid": "OnlyAllowFAS",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": "athena.amazonaws.com"
        }
      }
    }
  ]
}
```

Pour d'autres exemples d'utilisation de clés de condition pour autoriser l'accès à la FAS, consultez le référentiel data perimeter example policy repo.

Exemples de politiques basées sur l'identité IAM

Une politique est un objet AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit leurs autorisations. AWS évalue ces politiques lorsqu'un principal IAM (utilisateur ou rôle) fait une

demande. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée. La plupart des politiques sont stockées AWS sous forme de documents JSON attachés à une identité IAM (utilisateur, groupe d'utilisateurs ou rôle). Les politiques basées sur l'identité incluent des politiques gérées par AWS des politiques gérées par le client et des politiques en ligne. Pour apprendre à créer une politique IAM à l'aide de ces exemples de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

Par défaut, toutes les demandes sont refusées. Ainsi, vous devez fournir l'accès aux services, actions et ressources que vous avez l'intention d'autoriser en accès à l'identité. Si vous souhaitez également autoriser l'accès pour effectuer les actions spécifiées dans la console IAM, vous devez fournir des autorisations supplémentaires.

La bibliothèque de politiques suivante peut vous aider à définir des autorisations pour vos identités IAM. Une fois la politique recherchée trouvée, choisissez Afficher cette politique pour afficher le code JSON de la politique. Vous pouvez utiliser le document de politique JSON sous forme de modèle pour vos propres politiques.

> **Note**
>
> Si vous souhaitez envoyer une politique à inclure dans ce guide de référence, utilisez le bouton Commentaire situé au bas de cette page.

Exemples de politiques : AWS

- Autorise l'accès dans une plage de dates spécifique. (Afficher cette politique.)
- Permet d'activer et de désactiver AWS les régions. (Afficher cette politique.)
- Autorise les utilisateurs authentifiés par MFA à gérer leurs propres informations d'identification sur la page Mes informations d'identification de sécurité. (Afficher cette politique.)
- Autorise un accès spécifique en cas d'utilisation de MFA pendant une plage de dates spécifique. (Afficher cette politique.)
- Autorise les utilisateurs à gérer leurs propres informations d'identification sur la page Mes informations d'identification de sécurité. (Afficher cette politique.)
- Autorise les utilisateurs à gérer leur propre dispositif MFA sur la page Mes informations d'identification de sécurité. (Afficher cette politique.)
- Autorise les utilisateurs à gérer leur propre mot de passe sur la page Mes informations d'identification de sécurité. (Afficher cette politique.)

- Autorise les utilisateurs à gérer leurs propres mots de passe, clés d'accès et clés publiques SSH sur la page Mes informations d'identification de sécurité. (Afficher cette politique.)
- Refuse l'accès à AWS en fonction de la région demandée. (Afficher cette politique.)
- Refuse l'accès à AWS en fonction de l'adresse IP source. (Afficher cette politique.)

Exemple de politique : AWS Data Exchange

- Refuser l'accès aux ressources Amazon S3 en dehors de votre compte, sauf AWS Data Exchange. (Afficher cette politique.)

Exemples de politiques : AWS Data Pipeline

- Refuse l'accès aux pipelines qu'un utilisateur n'a pas créés (Afficher cette politique).

Exemple de politiques : Amazon DynamoDB

- Autorise l'accès à une table Amazon DynamoDB spécifique (afficher cette politique).
- Autorise l'accès à des attributs Amazon DynamoDB spécifiques (afficher cette politique).
- Autorise l'accès au niveau des éléments à Amazon DynamoDB en fonction d'un ID Amazon Cognito (Afficher cette politique).

Exemples de politiques : Amazon EC2

- Permet de joindre ou de détacher des volumes Amazon EBS à des EC2 instances Amazon en fonction de balises (consultez cette politique.)
- Permet de lancer EC2 des instances Amazon dans un sous-réseau spécifique, par programmation et dans la console (voir cette politique.)
- Permet de gérer les groupes EC2 de sécurité Amazon associés à un VPC spécifique, par programmation et dans la console (voir cette politique.)
- Permet de démarrer ou d'arrêter EC2 les instances Amazon qu'un utilisateur a balisées, par programmation et dans la console (voir cette politique.)
- Permet de démarrer ou d'arrêter EC2 des instances Amazon en fonction de la ressource et des balises principales, par programmation et dans la console (voir cette politique.)

- Permet de démarrer ou d'arrêter EC2 des instances Amazon lorsque la ressource et les balises principales correspondent (consultez cette politique.)
- Permet un EC2 accès complet à Amazon dans une région spécifique, par programmation et dans la console. (Afficher cette politique.)
- Permet de démarrer ou d'arrêter une EC2 instance Amazon spécifique et de modifier un groupe de sécurité spécifique, par programmation et dans la console (voir cette politique.)
- Refuse l'accès à des EC2 opérations Amazon spécifiques sans MFA (consultez cette politique.)
- Limite la résiliation des EC2 instances Amazon à une plage d'adresses IP spécifique (voir cette politique.)

Exemples de politiques : Gestion des identités et des accès AWS (IAM)

- Autorise l'accès à l'API du simulateur de politique (Afficher cette politique).
- Autorise l'accès à la console du simulateur de politique (Afficher cette politique).
- Permet d'endosser les rôles ayant une balise spécifique, par programmation et dans la console (Afficher cette politique).
- Autorise et refuse l'accès à plusieurs services, par programmation et dans la console (Afficher cette politique).
- Permet d'ajouter une balise spécifique à un utilisateur IAM avec une autre balise spécifique, par programmation et dans la console (afficher cette politique).
- Permet d'ajouter une balise spécifique à un utilisateur ou rôle IAM, par programmation et dans la console (afficher cette politique).
- Permet de créer un nouvel utilisateur uniquement avec des balises spécifiques (Afficher cette politique).
- Autorise la génération et l'extraction des rapports d'informations d'identification IAM (afficher cette politique).
- Autorise la gestion des membres d'un groupe, par programmation et dans la console (Afficher cette politique).
- Autorise la gestion d'une balise spécifique (Afficher cette politique).
- Autorise la transmission d'un rôle IAM à un service spécifique (afficher cette politique).
- Autorise l'accès en lecture seule à la console IAM sans notification (afficher cette politique).
- Autorise l'accès en lecture seule à la console IAM (afficher cette politique).

- Autorise des utilisateurs spécifiques à gérer un groupe, par programmation et dans la console (Afficher cette politique).
- Permet de définir les exigences de mot de passe de compte, par programmation et dans la console (Afficher cette politique).
- Autorise l'utilisation de l'API du simulateur de politiques pour les utilisateurs avec un chemin spécifique (Afficher cette politique).
- Autorise l'utilisation de la console du simulateur de politiques pour les utilisateurs avec un chemin spécifique (Afficher cette politique).
- Autorise les utilisateurs IAM à gérer eux-mêmes un dispositif MFA. (Afficher cette politique.)
- Autorise les utilisateurs IAM à définir leurs informations d'identification, par programmation et dans la console. (Afficher cette politique.)
- Permet d'afficher les informations du dernier accès au service pour une AWS Organizations politique dans la console IAM. (Afficher cette politique.)
- Limite les politiques gérées pouvant être appliquées à un nouvel utilisateur, groupe ou rôle IAM (afficher cette politique).
- Autorise l'accès aux politiques IAM uniquement dans votre compte (Afficher cette politique.)

Exemples de politiques : AWS Lambda

- Permet à une AWS Lambda fonction d'accéder à une table Amazon DynamoDB (consultez cette politique.)

Exemple de politiques : Amazon RDS

- Autorise l'accès complet à la base de données Amazon RDS dans une région donnée. (Afficher cette politique.)
- Autorise la restauration des bases de données Amazon RDS, par programmation et dans la console (afficher cette politique).
- Accorde aux propriétaires de balise l'accès complet aux ressources Amazon RDS qu'ils ont balisées (afficher cette politique).

Exemples de politiques : Amazon S3

- Autorise un utilisateur Amazon Cognito à accéder aux objets dans leur propre compartiment Amazon S3 (Afficher cette politique).
- Autorise un utilisateur avec des informations d'identification temporaires à accéder à leur répertoire d'accueil dans Amazon S3, par programmation et dans la console (Afficher cette politique)
- Autorise l'accès S3 complet, mais refuse explicitement l'accès au compartiment de production si l'administrateur ne s'est pas connecté à l'aide d'une autorisation MFA au cours des 30 dernières minutes (Afficher cette politique).
- Autorise les utilisateurs IAM à accéder à leurs répertoire d'accueil dans Amazon S3, par programmation et dans la console (Afficher cette politique)
- Permet à un utilisateur de gérer un seul compartiment Amazon S3 et refuse toutes les autres AWS actions et ressources (consultez cette politique.)
- Autorise l'accès en `Read` et en `Write` à un compartiment Amazon S3 spécifique (Afficher cette politique).
- Autorise l'accès en `Read` et en `Write` à un compartiment Amazon S3 spécifique, par programmation et dans la console (Afficher cette politique).

AWS : permet l'accès en fonction de la date et de l'heure

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès aux actions en fonction de la date et de l'heure. Cette politique limite l'accès aux actions effectuées entre le 1er avril 2020 et le 30 juin 2020 (UTC) inclus. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Pour en savoir plus sur l'utilisation de plusieurs conditions au sein du bloc `Condition` d'une politique IAM, consultez Plusieurs valeurs dans un élément Condition.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": "service-prefix:action-name",
            "Resource": "*",
            "Condition": {
                "DateGreaterThan": {"aws:CurrentTime": "2020-04-01T00:00:00Z"},
                "DateLessThan": {"aws:CurrentTime": "2020-06-30T23:59:59Z"}
            }
        }
    ]
}
```

> **Note**
>
> Vous ne pouvez pas utiliser une variable de politique avec l'opérateur de condition Date. Pour en savoir plus, consultez Élément de condition

AWS: permet d'activer et de désactiver AWS les régions

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un administrateur à activer et désactiver la région Asie-Pacifique (Hong Kong) (ap-east-1). Cette politique définit des autorisations pour l'accès à la console et par programmation. Ce paramètre s'affiche dans la page Paramètres du compte dans la Console de gestion AWS. Cette page comprend les informations de niveau compte sensibles qui doivent être affichées et gérées uniquement par les administrateurs de compte. Pour utiliser cette politique, remplacez celle de *italicized placeholder text* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

> **Important**
>
> Vous ne pouvez pas activer ou désactiver les régions qui sont activées par défaut. Vous pouvez uniquement inclure les régions désactivées par défaut. Pour plus d'informations, consultez Gestion des régions AWS dans le Références générales AWS.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "EnableDisableHongKong",
            "Effect": "Allow",
            "Action": [
                "account:EnableRegion",
                "account:DisableRegion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {"account:TargetRegion": "ap-east-1"}
            }
        },
        {
            "Sid": "ViewConsole",
            "Effect": "Allow",
            "Action": [
                "account:ListRegions"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS : autorise les utilisateurs IAM authentifiés par MFA à gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise les utilisateurs IAM authentifiés à l'aide de l'authentification multifactorielle (MFA) à gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité. Cette page Console de gestion AWS affiche les informations relatives au compte, telles que l'ID de compte et l'ID d'utilisateur canonique. Les utilisateurs peuvent également consulter et modifier leurs mots de passe, leurs clés d'accès, leurs dispositifs MFA, leurs certificats X.509, leurs clés SSH et leurs informations d'identification Git. Cet exemple de politique inclut les autorisations requises pour consulter et modifier toutes les informations de la page. Cela oblige également l'utilisateur à configurer et à

s'authentifier à l'aide de la MFA avant d'effectuer toute autre opération dans. AWS Pour autoriser les utilisateurs à gérer leurs propres informations d'identification sans utiliser l'authentification MFA, consultez AWS : autorise les utilisateurs IAM à gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

Pour en savoir plus sur comment les utilisateurs peuvent accéder à la page Informations d'identification de sécurité, consultez Comment les utilisateurs IAM modifient leur mot de passe (console).

> **Note**
>
> - Cet exemple de politique n'autorise pas les utilisateurs à réinitialiser un mot de passe lorsqu'ils se connectent Console de gestion AWS pour la première fois. Nous vous recommandons de n'accorder des autorisations aux nouveaux utilisateurs qu'après leur enregistrement. Pour de plus amples informations, veuillez consulter Comment créer des utilisateurs IAM en toute sécurité ?. Cela empêche également les utilisateurs dont le mot de passe a expiré de le réinitialiser pendant la procédure d'enregistrement. Vous pouvez autoriser cette opération en ajoutant `iam:ChangePassword` et `iam:GetAccountPasswordPolicy` à l'instruction `DenyAllExceptListedIfNoMFA`. Cependant, nous ne le recommandons pas, car autoriser les utilisateurs à modifier leur mot de passe sans MFA peut poser des problèmes de sécurité.
> - Si vous avez l'intention d'utiliser cette stratégie pour l'accès par programmation, vous devez appeler `GetSessionToken` pour une authetification auprès de MFA. Pour de plus amples informations, veuillez consulter Accès sécurisé aux API avec MFA.

À quoi sert cette politique ?

- L'instruction `AllowViewAccountInfo` permet à l'utilisateur d'afficher les informations au niveau du compte. Ces autorisations doivent se trouver dans leur propre instruction, car elles ne prennent pas en charge ou n'ont pas besoin de spécifier d'ARN de ressource particulier. À la place, les autorisations spécifient `"Resource" : "*"`. Cette instruction contient les actions suivantes qui permettent à l'utilisateur d'afficher des informations spécifiques :
  - `GetAccountPasswordPolicy` : afficher les exigences de mot de passe du compte lors de la modification de son propre mot de passe utilisateur IAM.
  - `ListVirtualMFADevices` : afficher les détails relatifs à un dispositif MFA virtuel qui est activé pour l'utilisateur.

- L'instruction `AllowManageOwnPasswords` permet à l'utilisateur de modifier son propre mot de passe. Cette instruction inclut également l'action `GetUser`, qui est requise pour afficher la plupart des informations de la page My security credentials (Mes informations d'identification de sécurité).
- L'instruction `AllowManageOwnAccessKeys` permet à l'utilisateur de créer, mettre à jour et supprimer ses propres clés d'accès. L'utilisateur récupère peut également récupérer des informations relatives au moment où la clé d'accès spécifiée a été utilisée pour la dernière fois.
- L'instruction `AllowManageOwnSigningCertificates` permet à l'utilisateur de charger, mettre à jour et supprimer ses propres certificats de signature.
- L'instruction `AllowManageOwnSSHPublicKeys` permet à l'utilisateur de charger, mettre à jour et supprimer ses propres clés SSH publiques pour CodeCommit.
- L'instruction `AllowManageOwnGitCredentials` permet à l'utilisateur de créer, mettre à jour et supprimer ses propres informations d'identification Git pour CodeCommit.
- L'instruction `AllowManageOwnVirtualMFADevice` permet à l'utilisateur de créer son propre dispositif MFA. L'ARN de ressource dans l'instruction autorise uniquement l'utilisateur à créer un dispositif MFA de n'importe quel nom, mais les autres instructions de la politique autorisent uniquement l'utilisateur à connecter le dispositif à l'utilisateur actuellement connecté.
- L'instruction `AllowManageOwnUserMFA` permet à l'utilisateur de consulter et gérer le dispositif MFA matériel, U2F ou virtuel pour leur propre utilisateur. L'ARN de ressource dans l'instruction autorise uniquement l'accès au propre utilisateur IAM de l'utilisateur. Les utilisateurs ne peuvent pas afficher ni gérer le dispositif MFA pour d'autres utilisateurs.
- La `DenyAllExceptListedIfNoMFA` déclaration refuse l'accès à toutes les actions dans tous les AWS services, à l'exception de quelques actions répertoriées, mais uniquement si l'utilisateur n'est pas connecté avec le MFA. L'instruction utilise une combinaison de `"Deny"` et `"NotAction"` pour refuser explicitement l'accès à toutes les actions non répertoriées. Les éléments répertoriés ne sont pas refusés ou autorisés par cette instruction. Toutefois, les actions sont autorisées par d'autres instructions de la politique. Pour plus d'informations sur la logique de cette instruction, voir NotAction avec Deny. Si l'utilisateur est connecté avec MFA, le test `Condition` échoue et cette instruction ne refuse aucune action. Dans ce cas, les autorisations de l'utilisateur sont définies par d'autres stratégies ou instructions.

  Cette instruction garantit à l'utilisateur de pouvoir effectuer uniquement les actions répertoriées lorsqu'il n'est pas connecté avec MFA. Par ailleurs, il peut effectuer les actions répertoriées uniquement si une autre instruction ou politique permet l'accès à ces actions. Cela ne permet pas à un utilisateur de créer un mot de passe lors de la connexion, car l'action `iam:ChangePassword` ne doit pas être autorisée sans l'authentification MFA.

La version `...IfExists` de l'opérateur `Bool` permet de s'assurer que si la clé `aws : MultiFactorAuthPresent` est manquante, la condition renvoie la valeur true. Cela signifie qu'un utilisateur qui accède à une API avec des informations d'identification à long terme, comme une clé d'accès, se voit refuser l'accès aux opérations d'API non IAM.

Cette politique ne permet pas aux utilisateurs d'afficher la page Users (Utilisateurs) dans la console IAM, ou d'utiliser cette page pour accéder à leurs propres informations utilisateur. Pour que cela soit possible, ajoutez l'action `iam:ListUsers` à l'instruction `AllowViewAccountInfo` et à l'instruction `DenyAllExceptListedIfNoMFA`. Cette stratégie ne permet pas non plus aux utilisateurs de modifier leur mot de passe sur leur propre page utilisateur. Pour que cela soit possible, ajoutez les actions `iam:GetLoginProfile` et `iam:UpdateLoginProfile` à l'instruction `AllowManageOwnPasswords`. Par ailleurs, pour autoriser un utilisateur à modifier son mot de passe dans sa propre page utilisateur sans se connecter à l'aide de MFA, ajoutez l'action `iam:UpdateLoginProfile` à l'instruction `DenyAllExceptListedIfNoMFA`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
```

```
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSigningCertificates",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSigningCertificate",
                "iam:ListSigningCertificates",
                "iam:UpdateSigningCertificate",
                "iam:UploadSigningCertificate"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnGitCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:ResetServiceSpecificCredential",
                "iam:UpdateServiceSpecificCredential"
            ],
```

```
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice",
                "sts:GetSessionToken"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

AWS : autorise l'accès spécifique dans un délai spécifique à l'aide de MFA

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui utilise plusieurs conditions qui sont évaluées à l'aide d'un opérateur logique `AND`. Il permet l'accès complet au service nommé `SERVICE-NAME-1`, et l'accès aux actions `ACTION-NAME-A` et `ACTION-NAME-B` dans le service nommé `SERVICE-NAME-2`. Ces actions sont autorisées uniquement lorsque l'utilisateur est authentifié à l'aide de l'authentification multifacteur (Multi-Factor Authentication, MFA). L'accès est limité aux actions se produisant entre le 1er juillet 2017 et le 31 décembre 2017 (UTC) inclus. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Pour en savoir plus sur l'utilisation de plusieurs conditions au sein du bloc `Condition` d'une politique IAM, consultez Plusieurs valeurs dans un élément Condition.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "service-prefix-1:*",
            "service-prefix-2:action-name-a",
            "service-prefix-2:action-name-b"
        ],
        "Resource": "*",
        "Condition": {
            "Bool": {"aws:MultiFactorAuthPresent": true},
            "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
            "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
        }
    }
}
```

AWS : autorise les utilisateurs IAM à gérer leurs propres informations d’identification sur la page Informations d’identification de sécurité

Cet exemple montre comment vous pouvez créer une politique basée sur l’identité qui autorise des utilisateurs IAM à gérer leurs propres informations d’identification sur la page Informations d’identification de sécurité. Cette Console de gestion AWS page affiche les informations du compte, telles que l'identifiant du compte et l'identifiant utilisateur canonique. Les utilisateurs peuvent également consulter et modifier leurs propres mots de passe, clés d'accès, certificats X.509, clés SSH et informations d'identification Git. Cet exemple de politique inclut les autorisations requises pour consulter et modifier toutes les informations de la page sauf le dispositif MFA de l'utilisateur. Pour autoriser les utilisateurs à gérer leurs propres informations d'identification avec l'authentification MFA, consultez AWS : autorise les utilisateurs IAM authentifiés par MFA à gérer leurs propres informations d’identification sur la page Informations d’identification de sécurité.

Pour en savoir plus sur comment les utilisateurs peuvent accéder à la page Informations d’identification de sécurité, consultez Comment les utilisateurs IAM modifient leur mot de passe (console).

À quoi sert cette politique ?

- L'instruction `AllowViewAccountInfo` permet à l'utilisateur d'afficher les informations au niveau du compte. Ces autorisations doivent se trouver dans leur propre instruction, car elles ne prennent pas en charge ou n'ont pas besoin de spécifier d'ARN de ressource particulier. À la place, les autorisations spécifient `"Resource" : "*"`. Cette instruction contient les actions suivantes qui permettent à l'utilisateur d'afficher des informations spécifiques :
  - `GetAccountPasswordPolicy` : afficher les exigences de mot de passe du compte lors de la modification de son propre mot de passe utilisateur IAM.
  - `GetAccountSummary` : afficher l'ID du compte et l'ID d'utilisateur canonique du compte.
- L'instruction `AllowManageOwnPasswords` permet à l'utilisateur de modifier son propre mot de passe. Cette instruction inclut également l'action `GetUser`, qui est requise pour afficher la plupart des informations de la page My security credentials (Mes informations d'identification de sécurité).
- L'instruction `AllowManageOwnAccessKeys` permet à l'utilisateur de créer, mettre à jour et supprimer ses propres clés d'accès. L'utilisateur récupère peut également récupérer des informations relatives au moment où la clé d'accès spécifiée a été utilisée pour la dernière fois.
- L'instruction `AllowManageOwnSigningCertificates` permet à l'utilisateur de charger, mettre à jour et supprimer ses propres certificats de signature.

- L'instruction `AllowManageOwnSSHPublicKeys` permet à l'utilisateur de charger, mettre à jour et supprimer ses propres clés SSH publiques pour CodeCommit.
- L'instruction `AllowManageOwnGitCredentials` permet à l'utilisateur de créer, mettre à jour et supprimer ses propres informations d'identification Git pour CodeCommit.

Cette politique ne permet pas aux utilisateurs de consulter ni de gérer leurs propres dispositifs MFA. Ils ne peuvent pas afficher la page Users (Utilisateurs) dans la console IAM, ou utiliser cette page pour accéder à leurs propres informations utilisateur. Pour que cela soit possible, ajoutez l'action `iam:ListUsers` à l'instruction `AllowViewAccountInfo`. Cette stratégie ne permet pas non plus aux utilisateurs de modifier leur mot de passe sur leur propre page utilisateur. Pour que cela soit possible, ajoutez les actions `iam:CreateLoginProfile`, `iam:DeleteLoginProfile`, `iam:GetLoginProfile` et `iam:UpdateLoginProfile` à l'instruction `AllowManageOwnPasswords`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
```

```
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSigningCertificates",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSigningCertificate",
                "iam:ListSigningCertificates",
                "iam:UpdateSigningCertificate",
                "iam:UploadSigningCertificate"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnGitCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:ResetServiceSpecificCredential",
                "iam:UpdateServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
```

```
    ]
}
```

AWS : autorise les utilisateurs IAM authentifiés par MFA à gérer leur dispositif MFA sur la page Informations d'identification de sécurité

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise les utilisateurs IAM authentifiés à l'aide de l'authentification multifactorielle (MFA) à gérer leur propre dispositif MFA sur la page Informations d'identification de sécurité. Cette Console de gestion AWS page affiche les informations relatives au compte et à l'utilisateur, mais l'utilisateur ne peut consulter et modifier que son propre dispositif MFA. Pour autoriser les utilisateurs à gérer toutes leurs propres informations d'identification avec l'authentification MFA, consultez AWS : autorise les utilisateurs IAM authentifiés par MFA à gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

> **Note**
>
> Si un utilisateur IAM doté de cette politique n'est pas authentifié par MFA, cette politique refuse l'accès à toutes les AWS actions, à l'exception de celles nécessaires pour s'authentifier à l'aide de l'authentification MFA. Pour utiliser l' AWS API AWS CLI et, les utilisateurs IAM doivent d'abord récupérer leur jeton MFA à l'aide de AWS STS GetSessionTokenl'opération, puis utiliser ce jeton pour authentifier l'opération souhaitée. D'autres politiques, telles que les politiques basées sur les ressources ou d'autres politiques basées sur l'identité peuvent autoriser des actions dans d'autres services. Cette politique refusera cet accès si l'utilisateur IAM n'est pas authentifié avec MFA.

Pour en savoir plus sur comment les utilisateurs peuvent accéder à la page Informations d'identification de sécurité, consultez Comment les utilisateurs IAM modifient leur mot de passe (console).

À quoi sert cette politique ?

- L'instruction `AllowViewAccountInfo` permet à l'utilisateur d'afficher les détails relatifs à un dispositif MFA virtuel, qui est activé pour l'utilisateur. Cette autorisation doit se trouver dans sa propre instruction, car elle ne prend pas en charge la spécification d'un ARN de ressource. À la place, vous devez spécifier `"Resource" : "*"`.

- L'instruction `AllowManageOwnVirtualMFADevice` permet à l'utilisateur de créer son propre dispositif MFA. L'ARN de ressource dans l'instruction autorise uniquement l'utilisateur à créer un dispositif MFA de n'importe quel nom, mais les autres instructions de la politique autorisent uniquement l'utilisateur à connecter le dispositif à l'utilisateur actuellement connecté.
- L'instruction `AllowManageOwnUserMFA` permet à l'utilisateur de consulter et gérer son propre dispositif MFA matériel, U2F ou virtuel. L'ARN de ressource dans l'instruction autorise uniquement l'accès au propre utilisateur IAM de l'utilisateur. Les utilisateurs ne peuvent pas afficher ni gérer le dispositif MFA pour d'autres utilisateurs.
- La `DenyAllExceptListedIfNoMFA` déclaration refuse l'accès à toutes les actions dans tous les AWS services, à l'exception de quelques actions répertoriées, mais uniquement si l'utilisateur n'est pas connecté avec le MFA. L'instruction utilise une combinaison de `"Deny"` et `"NotAction"` pour refuser explicitement l'accès à toutes les actions non répertoriées. Les éléments répertoriés ne sont pas refusés ou autorisés par cette instruction. Toutefois, les actions sont autorisées par d'autres instructions de la politique. Pour plus d'informations sur la logique de cette instruction, consultez la section NotAction avec Deny. Si l'utilisateur est connecté avec MFA, le test `Condition` échoue et cette instruction ne refuse aucune action. Dans ce cas, les autorisations de l'utilisateur sont définies par d'autres stratégies ou instructions.

  Cette instruction garantit à l'utilisateur de pouvoir effectuer uniquement les actions répertoriées lorsqu'il n'est pas connecté avec MFA. Par ailleurs, il peut effectuer les actions répertoriées uniquement si une autre instruction ou politique permet l'accès à ces actions.

  La version `...IfExists` de l'opérateur `Bool` permet de s'assurer que si la clé `aws:MultiFactorAuthPresent` est manquante, la condition renvoie la valeur true. Cela signifie qu'un utilisateur qui accède à une opération d'API avec des informations d'identification à long terme, comme une clé d'accès, se voit refuser l'accès aux opérations d'API non IAM.

Cette politique ne permet pas aux utilisateurs d'afficher la page Users (Utilisateurs) dans la console IAM, ou d'utiliser cette page pour accéder à leurs propres informations utilisateur. Pour que cela soit possible, ajoutez l'action `iam:ListUsers` à l'instruction `AllowViewAccountInfo` et à l'instruction `DenyAllExceptListedIfNoMFA`.

> **Warning**
>
> N'ajoutez pas d'autorisations pour supprimer un dispositif MFA sans authentification MFA. Les utilisateurs dotés de cette politique peuvent tenter de s'attribuer un dispositif MFA virtuel et recevoir une erreur indiquant qu'ils ne sont pas autorisés à effectuer

`iam:DeleteVirtualMFADevice`. Si cela se produit, n'ajoutez pas cette autorisation à l'instruction `DenyAllExceptListedIfNoMFA`. Les utilisateurs qui ne sont pas authentifiés avec MFA ne doivent jamais être autorisés à supprimer un dispositif MFA. Les utilisateurs peuvent consulter cette erreur s'ils ont précédemment commencé à affecter un dispositif MFA virtuel à leur utilisateur et annulé le processus. Pour résoudre ce problème, vous ou un autre administrateur devez supprimer le dispositif MFA virtuel existant de l'utilisateur à l'aide de l'API AWS CLI or AWS . Pour de plus amples informations, veuillez consulter Je ne suis pas autorisé à effectuer : iam : DeleteVirtualMFADevice.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": "iam:ListVirtualMFADevices",
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
```

```
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice",
                "sts:GetSessionToken"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}
            }
        }
    ]
}
```

AWS : autorise les utilisateurs IAM à modifier leur propre mot de passe pour la console sur la page Informations d'identification de sécurité

Cet exemple montre comment créer une politique basée sur l'identité qui permet aux utilisateurs IAM de modifier leur propre Console de gestion AWS mot de passe sur la page Informations d'identification de sécurité. Cette Console de gestion AWS page affiche les informations relatives au compte et à l'utilisateur, mais l'utilisateur ne peut accéder qu'à son propre mot de passe. Pour autoriser les utilisateurs à gérer toutes leurs propres informations d'identification avec l'authentification MFA, consultez AWS : autorise les utilisateurs IAM authentifiés par MFA à gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité. Pour autoriser les utilisateurs à gérer leurs propres informations d'identification sans utiliser l'authentification MFA, consultez AWS : autorise les utilisateurs IAM à gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

Pour en savoir plus sur comment les utilisateurs peuvent accéder à la page Informations d'identification de sécurité, consultez Comment les utilisateurs IAM modifient leur mot de passe (console).

À quoi sert cette politique ?

- L'instruction `ViewAccountPasswordRequirements` autorise l'utilisateur à afficher les exigences de mot de passe du compte lors de la modification de son propre mot de passe utilisateur IAM.
- L'instruction `ChangeOwnPassword` permet à l'utilisateur de modifier son propre mot de passe. Cette instruction inclut également l'action `GetUser`, qui est requise pour afficher la plupart des informations de la page My security credentials (Mes informations d'identification de sécurité).

Cette politique ne permet pas aux utilisateurs d'afficher la page Users (Utilisateurs) dans la console IAM, ou d'utiliser cette page pour accéder à leurs propres informations utilisateur. Pour que cela soit possible, ajoutez l'action `iam:ListUsers` à l'instruction `ViewAccountPasswordRequirements`. Cette stratégie ne permet pas non plus aux utilisateurs de modifier leur mot de passe sur leur propre page utilisateur. Pour que cela soit possible, ajoutez les actions `iam:GetLoginProfile` et `iam:UpdateLoginProfile` à l'instruction `ChangeOwnPassword`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ViewAccountPasswordRequirements",
            "Effect": "Allow",
            "Action": "iam:GetAccountPasswordPolicy",
            "Resource": "*"
        },
        {
            "Sid": "ChangeOwnPassword",
            "Effect": "Allow",
            "Action": [
                "iam:GetUser",
                "iam:ChangePassword"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS : autorise les utilisateurs IAM à gérer leurs propres mot de passe, clés d’accès et clés publiques SSH sur la page Informations d’identification de sécurité

Cet exemple montre comment vous pouvez créer une politique basée sur l’identité qui autorise des utilisateurs IAM à gérer leurs propres mot de passe, clés d’accès et certificats X.509 sur la page Informations d’identification de sécurité. Cette page Console de gestion AWS affiche les informations relatives au compte, telles que l'ID de compte et l'ID d'utilisateur canonique. Les utilisateurs peuvent également consulter et modifier leurs propres mots de passe, clés d'accès, dispositifs MFA, certificats X.509, clés SSH et informations d'identification Git. Cet exemple de politique inclut les autorisations requises pour consulter et modifier uniquement le mot de passe, les clés d'accès et le certificat X.509. Pour autoriser les utilisateurs à gérer toutes leurs propres informations d'identification avec l'authentification MFA, consultez AWS : autorise les utilisateurs IAM authentifiés par MFA à gérer leurs propres informations d’identification sur la page Informations d’identification de sécurité. Pour autoriser les utilisateurs à gérer leurs propres informations d'identification sans utiliser l'authentification MFA, consultez AWS : autorise les utilisateurs IAM à gérer leurs propres informations d’identification sur la page Informations d’identification de sécurité.

Pour en savoir plus sur comment les utilisateurs peuvent accéder à la page Informations d’identification de sécurité, consultez Comment les utilisateurs IAM modifient leur mot de passe (console).

À quoi sert cette politique ?

- L'instruction `AllowViewAccountInfo` permet à l'utilisateur d'afficher les informations au niveau du compte. Ces autorisations doivent se trouver dans leur propre instruction, car elles ne prennent pas en charge ou n'ont pas besoin de spécifier d'ARN de ressource particulier. À la place, les autorisations spécifient `"Resource" : "*"`. Cette instruction contient les actions suivantes qui permettent à l'utilisateur d'afficher des informations spécifiques :
  - `GetAccountPasswordPolicy` : afficher les exigences de mot de passe du compte lors de la modification de son propre mot de passe utilisateur IAM.
  - `GetAccountSummary` : afficher l'ID du compte et l'ID d'utilisateur canonique du compte.
- L'instruction `AllowManageOwnPasswords` permet à l'utilisateur de modifier son propre mot de passe. Cette instruction inclut également l'action `GetUser`, qui est requise pour afficher la plupart des informations de la page My security credentials (Mes informations d'identification de sécurité).
- L'instruction `AllowManageOwnAccessKeys` permet à l'utilisateur de créer, mettre à jour et supprimer ses propres clés d'accès. L'utilisateur récupère peut également récupérer des informations relatives au moment où la clé d'accès spécifiée a été utilisée pour la dernière fois.

- L'instruction `AllowManageOwnSSHPublicKeys` permet à l'utilisateur de charger, mettre à jour et supprimer ses propres clés SSH publiques pour CodeCommit.

Cette politique ne permet pas aux utilisateurs de consulter ni de gérer leurs propres dispositifs MFA. Ils ne peuvent pas afficher la page Users (Utilisateurs) dans la console IAM, ou utiliser cette page pour accéder à leurs propres informations utilisateur. Pour que cela soit possible, ajoutez l'action `iam:ListUsers` à l'instruction `AllowViewAccountInfo`. Cette stratégie ne permet pas non plus aux utilisateurs de modifier leur mot de passe sur leur propre page utilisateur. Pour que cela soit possible, ajoutez les actions `iam:GetLoginProfile` et `iam:UpdateLoginProfile` à l'instruction `AllowManageOwnPasswords`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
```

```
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: refuse l'accès AWS en fonction de la région demandée

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui refuse l'accès à toutes les actions en dehors des régions spécifiées avec la clé de condition `aws:RequestedRegion`, à l'exception des actions dans les services spécifiés avec `NotAction`. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Cette politique utilise l'élément `NotAction` avec l'effet `Deny`, qui refuse explicitement l'accès à toutes les actions non répertoriées dans l'instruction. Les actions relatives à l' CloudFrontIAM, à Route 53 et Support aux services ne doivent pas être refusées, car il s'agit de services AWS mondiaux populaires dotés d'un point de terminaison unique situé physiquement dans la `us-east-1` région. Étant donné que toutes les demandes adressées à ces services sont effectuées vers la région `us-east-1`, les demandes sont refusées sans l'élément `NotAction`. Modifiez cet élément pour inclure les actions pour d'autres services AWS globaux que vous utilisez, tels que `budgets`, `globalaccelerator`, `importexport`, `organizations` ou `waf`. Certains autres services internationaux, tels que Amazon Q Developer dans les applications de chat AWS Device Farm, sont des services mondiaux dont les points de terminaison sont physiquement situés dans la `us-west-2`

région. Pour en savoir plus sur tous les services qui ont un seul point de terminaison global, consultez AWS Régions et points de terminaison dans le document Références générales AWS. Pour de plus amples informations sur l'utilisation de l'élément `NotAction` avec l'effet `Deny`, veuillez consulter Éléments de politique JSON IAM : NotAction.

> **Important**
>
> Cette politique ne permet aucune action. Utilisez cette stratégie conjointement à d'autres stratégies qui autorisent des actions spécifiques.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "DenyAllOutsideRequestedRegions",
            "Effect": "Deny",
            "NotAction": [
                "cloudfront:*",
                "iam:*",
                "organizations:*",
                "route53:*",
                "support:*"
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestedRegion": [
                        "eu-central-1",
                        "eu-west-1",
                        "eu-west-2",
                        "eu-west-3"
                    ]
                }
            }
        }
    ]
}
```

AWS: refuse l'accès à la AWS base de l'adresse IP source

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui refuse l'accès à toutes les AWS actions du compte lorsque la demande provient de principaux situés en dehors de la plage d'adresses IP spécifiée. Cette politique est utile lorsque les adresses IP de votre entreprise se situent dans les plages d'adresses IP spécifiées. Dans cet exemple, la demande sera rejetée sauf si elle provient de la plage CIDR 192.0.2.0/24 ou 203.0.113.0/24. La politique ne refuse pas les demandes effectuées par les AWS services utilisateurs, Transmission des sessions d'accès car l'adresse IP du demandeur d'origine est préservée.

Soyez prudent lorsque vous utilisez des conditions négatives dans la même instruction de politique comme `"Effect": "Deny"`. Dans ce cas, les actions spécifiées dans l'instruction de politique sont explicitement refusées dans toutes les conditions sauf celles spécifiées.

> **Important**
>
> Cette politique ne permet aucune action. Utilisez cette stratégie conjointement à d'autres stratégies qui autorisent des actions spécifiques.

Lorsque d'autres politiques autorisent des actions, les principaux peuvent effectuer des demandes à partir de la plage d'adresses IP. Un AWS service peut également effectuer des demandes en utilisant les informations d'identification du principal. Lorsqu'un principal effectue une demande en dehors de la plage d'adresses IP, la demande est refusée.

Pour de plus amples informations sur l'utilisation de la clé de condition `aws:SourceIp`, en particulier sur le moment où `aws:SourceIp` risque de ne pas fonctionne dans votre politique, veuillez consulter AWS clés de contexte de condition globale.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
            "NotIpAddress": {
```

```
                "aws:SourceIp": [
                    "192.0.2.0/24",
                    "203.0.113.0/24"
                ]
            }
        }
    }
}
```

AWS: Refusez l'accès aux ressources Amazon S3 en dehors de votre compte, sauf AWS Data Exchange

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui refuse l'accès à toutes les ressources AWS qui n'appartiennent pas à votre compte, à l'exception des ressources nécessaires au fonctionnement normal. AWS Data Exchange Pour utiliser cette politique, remplacez celle de *italicized placeholder text* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Vous pouvez créer une politique similaire pour restreindre l'accès aux ressources au sein d'une organisation ou d'une unité organisationnelle, tout en comptabilisant les ressources AWS Data Exchange détenues à l'aide des clés de condition `aws:ResourceOrgPaths` et`aws:ResourceOrgID`.

Si vous l'utilisez AWS Data Exchange dans votre environnement, le service crée et interagit avec des ressources telles que les compartiments Amazon S3 appartenant au compte de service. Par exemple, AWS Data Exchange envoie des demandes aux compartiments Amazon S3 appartenant au AWS Data Exchange service pour le compte du principal IAM (utilisateur ou rôle) invoquant le. AWS Data Exchange APIs Dans ce cas `aws:ResourceAccount`aws:ResourceOrgPaths, l'utilisation ou l'`aws:ResourceOrgID`application d'une politique sans tenir compte des ressources AWS Data Exchange détenues empêche l'accès aux compartiments appartenant au compte de service.

- Instruction `DenyAllAwsResourcesOutsideAccountExceptS3` utilise le l'élément `NotAction` avec l'effet Deny (Refuser) qui refuse explicitement l'accès à toutes les actions non répertoriées dans l'instruction et qui n'appartiennent pas non plus au compte répertorié. L'élément `NotAction` indique les exceptions à cette instruction. Ces actions constituent une exception à cette déclaration, car si elles sont effectuées sur des ressources créées par AWS Data Exchange, la politique les refuse.

- L'instruction `DenyAllS3ResourcesOutsideAccountExceptDataExchange` utilise une combinaison des clés de condition `ResourceAccount` et `CalledVia` pour refuser l'accès aux trois actions Amazon S3 exclues dans la déclaration précédente. L'instruction refuse les actions si les ressources n'appartiennent pas au compte répertorié et si le service appelant n'est pas AWS Data Exchange. L'instruction ne refuse pas les actions si la ressource appartient au compte répertorié ou si le principal de service répertorié, `dataexchange.amazonaws.com`, effectue les opérations.

> **Important**
>
> Cette politique ne permet aucune action. Elle utilise l'effet `Deny`, qui refuse explicitement l'accès à toutes les ressources répertoriées dans l'instruction n'appartenant pas au compte répertorié. Utilisez cette politique en combinaison avec d'autres politiques qui autorisent l'accès à des ressources spécifiques.

L'exemple suivant montre comment vous pouvez configurer la politique pour autoriser l'accès aux compartiments Amazon S3 requis.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAllAwsReourcesOutsideAccountExceptAmazonS3",
      "Effect": "Deny",
      "NotAction": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
```

```
      }
    },
    {
      "Sid": "DenyAllS3ResourcesOutsideAccountExceptDataExchange",
      "Effect": "Deny",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        },
        "ForAllValues:StringNotEquals": {
          "aws:CalledVia": [
            "dataexchange.amazonaws.com"
          ]
        }
      }
    }
  ]
}
```

AWS Data Pipeline: refuse l'accès aux DataPipeline pipelines qu'un utilisateur n'a pas créés

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui refuse l'accès aux pipelines qu'un utilisateur n'a pas créés. Si la valeur du champ `PipelineCreator` correspond au nom d'utilisateur IAM, alors les actions spécifiées ne sont pas refusées. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI

> **Important**
>
> Cette politique ne permet aucune action. Utilisez cette stratégie conjointement à d'autres stratégies qui autorisent des actions spécifiques.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ExplicitDenyIfNotTheOwner",
            "Effect": "Deny",
            "Action": [
                "datapipeline:ActivatePipeline",
                "datapipeline:AddTags",
                "datapipeline:DeactivatePipeline",
                "datapipeline:DeletePipeline",
                "datapipeline:DescribeObjects",
                "datapipeline:EvaluateExpression",
                "datapipeline:GetPipelineDefinition",
                "datapipeline:PollForTask",
                "datapipeline:PutPipelineDefinition",
                "datapipeline:QueryObjects",
                "datapipeline:RemoveTags",
                "datapipeline:ReportTaskProgress",
                "datapipeline:ReportTaskRunnerHeartbeat",
                "datapipeline:SetStatus",
                "datapipeline:SetTaskStatus",
                "datapipeline:ValidatePipelineDefinition"
            ],
            "Resource": ["*"],
            "Condition": {
                "StringNotEquals": {"datapipeline:PipelineCreator":
 "${aws:userid}"}
            }
        }
    ]
}
```

Amazon DynamoDB : autorise l'accès à une table spécifique

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès complet à la table DynamoDB `MyTable`. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par

vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

> **Important**
>
> Cette politique autorise toutes les actions pouvant être effectuées sur une table DynamoDB. Pour vérifier ces actions, consultez Autorisations d'API Amazon DynamoDB : référence des actions, ressources et conditions dans le Guide du développeur Amazon DynamoDB. Vous pouvez fournir les mêmes autorisations en répertoriant chaque action individuelle. Toutefois, si vous utilisez le caractère générique (*) dans l'élément `Action`, par exemple `"dynamodb:List*"`, vous n'avez pas besoin de mettre à jour votre politique si ajoute une nouvelle action Liste.

Cette politique autorise les actions DynamoDB uniquement sur les tables existantes et dont le nom est spécifié. Pour permettre à vos utilisateurs `Read` d'accéder à tous les éléments de DynamoDB, vous pouvez également joindre AmazonDynamoDBReadOnlyAccess AWS la politique gérée.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListAndDescribe",
            "Effect": "Allow",
            "Action": [
                "dynamodb:List*",
                "dynamodb:DescribeReservedCapacity*",
                "dynamodb:DescribeLimits",
                "dynamodb:DescribeTimeToLive"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SpecificTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb:BatchGet*",
                "dynamodb:DescribeStream",
```

```
                "dynamodb:DescribeTable",
                "dynamodb:Get*",
                "dynamodb:Query",
                "dynamodb:Scan",
                "dynamodb:BatchWrite*",
                "dynamodb:CreateTable",
                "dynamodb:Delete*",
                "dynamodb:Update*",
                "dynamodb:PutItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/MyTable"
        }
    ]
}
```

Amazon DynamoDB : autorise l'accès à des attributs spécifiques

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès aux attributs DynamoDB spécifiques. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *italicized placeholder text* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

L'exigence `dynamodb:Select` empêche l'action d'API de retourner les attributs qui ne sont pas autorisés, par exemple à partir d'une projection d'index. Pour en savoir plus sur les clés de condition DynamoDB, consultez Spécification de conditions : utilisation de clés de condition dans le Guide du développeur Amazon DynamoDB. Pour en savoir plus sur l'utilisation de plusieurs conditions ou de plusieurs clés de condition dans le `Condition` bloc d'une politique IAM, consultez Plusieurs valeurs dans un élément Condition.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:GetItem",
```

```
                "dynamodb:BatchGetItem",
                "dynamodb:Query",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem",
                "dynamodb:DeleteItem",
                "dynamodb:BatchWriteItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/table-name"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:Attributes": [
                        "column-name-1",
                        "column-name-2",
                        "column-name-3"
                    ]
                },
                "StringEqualsIfExists": {"dynamodb:Select":
 "SPECIFIC_ATTRIBUTES"}
            }
        }
    ]
}
```

Amazon DynamoDB : autorise l'accès au niveau de l'élément à DynamoDB en fonction d'un ID Amazon Cognito

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès au niveau de l'élément à la table DynamoDB `MyTable` en fonction de l'ID d'utilisateur de la réserve d'identités Amazon Cognito. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Pour utiliser cette politique, vous devez structurer votre table DynamoDB pour que l'ID d'utilisateur de la réserve d'identités Amazon Cognito soit la clé de partition. Pour plus d'informations, consultez Création d'une table dans le Guide du développeur Amazon DynamoDB.

Pour en savoir plus sur les clés de condition DynamoDB, consultez Spécification de conditions : utilisation de clés de condition dans le Guide du développeur Amazon DynamoDB.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:Query",
                "dynamodb:UpdateItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/MyTable"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": ["${cognito-
identity.amazonaws.com:sub}"]
                }
            }
        }
    ]
}
```

Amazon EC2 : attacher ou détacher des volumes Amazon EBS aux instances EC2 en fonction des balises

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise les propriétaires de volume EBS à attacher ou détacher leurs volumes EBS définis à l'aide de la balise `VolumeUser` aux instances EC2 qui sont balisées en tant qu'instances de développement (`Department=Development`). Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Pour de plus amples informations sur la création de politiques IAM pour contrôler l'accès aux ressources Amazon EC2, consultez Contrôle de l'accès aux ressources Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/Department": "Development"}
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:*:*:volume/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/VolumeUser": "${aws:username}"}
            }
        }
    ]
}
```

Amazon EC2 : autorise le lancement d'instances EC2 dans un sous-réseau spécifique, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui permet de répertorier les informations de tous les objets EC2 et de lancer des instances EC2 dans un sous-réseau spécifique. Cette politique définit des autorisations pour l'accès à la console et par

programmation. Pour utiliser cette politique, remplacez celle de *italicized placeholder text* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:GetConsole*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/subnet-subnet-id",
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ec2:*:*:volume/*",
                "arn:aws:ec2:*::image/ami-*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:security-group/*"
            ]
        }
    ]
}
```

Amazon EC2 : autorise la gestion des groupes de sécurité EC2 avec une paire spécifique d'étiquettes de valeur clé de manière programmatique et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui accorde aux utilisateurs l'autorisation de prendre certaines mesures pour les groupes de sécurité qui ont la même étiquette. Cette politique accorde des autorisations pour afficher les groupes de sécurité dans la console Amazon EC2, ajouter et supprimer des règles entrantes et sortantes, et répertorier

et modifier les descriptions de règles pour les groupes de sécurité existants portant l'étiquette `Department=Test`. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez celle de *italicized placeholder text* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeSecurityGroups",
         "ec2:DescribeSecurityGroupRules",
         "ec2:DescribeTags"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:AuthorizeSecurityGroupIngress",
         "ec2:RevokeSecurityGroupIngress",
         "ec2:AuthorizeSecurityGroupEgress",
         "ec2:RevokeSecurityGroupEgress",
         "ec2:ModifySecurityGroupRules",
         "ec2:UpdateSecurityGroupRuleDescriptionsIngress",
         "ec2:UpdateSecurityGroupRuleDescriptionsEgress"
      ],
      "Resource": [
         "arn:aws:ec2:us-east-1:111122223333:security-group/*"
      ],
      "Condition": {
         "StringEquals": {
            "aws:ResourceTag/Department": "Test"
         }
      }
     },
     {
      "Effect": "Allow",
      "Action": [
```

```
            "ec2:ModifySecurityGroupRules"
        ],
        "Resource": [
            "arn:aws:ec2:us-east-1:111122223333:security-group-rule/*"
        ]
      }
   ]
}
```

Amazon EC2 : autorise le démarrage ou l'arrêt des instances EC2 balisées par un utilisateur, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un utilisateur IAM à démarrer ou arrêter des instances EC2, mais uniquement si la balise d'instance `Owner` a la valeur du nom d'utilisateur de cet utilisateur. Cette politique définit des autorisations pour l'accès à la console et par programmation.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Owner": "${aws:username}"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
```

```
}
```

EC2: démarrer ou arrêter des instances en fonction des balises

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise le démarrage ou l'arrêt d'instances avec la paire clé-valeur de balise `Project = DataAnalytics`, mais uniquement par les principaux avec la paire clé-valeur de balise `Department = Data`. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

La condition de la politique renvoie la valeur true si les deux parties de la condition sont true. L'instance doit posséder la balise `Project=DataAnalytics`. En outre, le principal IAM (utilisateur ou rôle) qui fait la demande doit posséder la balise `Department=Data`.

> **Note**
>
> À titre de bonne pratique, associez les politiques avec la clé de condition `aws:PrincipalTag` aux groupes IAM pour le cas où certains utilisateurs pourraient avoir la balise spécifiée et d'autres pas.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "StartStopIfTags",
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:us-east-1:123456789012:instance/*",
            "Condition": {
                "StringEquals": {
```

```
                    "aws:ResourceTag/Project": "DataAnalytics",
                    "aws:PrincipalTag/Department": "Data"
                }
            }
        }
    ]
}
```

EC2 : démarrer ou arrêter les instances en fonction des balises de ressources et de principaux correspondantes

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un principal à démarrer ou arrêter une instance Amazon EC2 lorsque la balise de ressource de l'instance et la balise du principal ont la même valeur pour la clé de balise `CostCenter`. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

> **Note**
>
> À titre de bonne pratique, associez les politiques avec la clé de condition `aws:PrincipalTag` aux groupes IAM pour le cas où certains utilisateurs pourraient avoir la balise spécifiée et d'autres pas.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "ec2:startInstances",
            "ec2:stopInstances"
        ],
        "Resource": "*",
        "Condition": {"StringEquals":
            {"aws:ResourceTag/CostCenter": "${aws:PrincipalTag/CostCenter}"}}
```

```
    }
}
```

Amazon EC2 : autorise l'accès EC2 complet dans une région spécifique, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un accès EC2 total dans une région spécifique. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez celle de *italicized placeholder text* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique). Pour obtenir la liste des codes de région, veuillez consulter Régions disponibles dans le Guide de l'utilisateur Amazon EC2.

Sinon, vous pouvez utiliser la clé de condition générale `aws:RequestedRegion`, qui est prise en charge par toutes les actions d'API Amazon EC2. Pour de plus amples informations, veuillez consulter la section Exemple : restriction de l'accès à une région spécifique dans le Guide de l'utilisateur d'Amazon EC2.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": "ec2:*",
            "Resource": "*",
            "Effect": "Allow",
            "Condition": {
                "StringEquals": {
                    "ec2:Region": "us-east-2"
                }
            }
        }
    ]
}
```

Amazon EC2 : autorise le démarrage ou l'arrêt d'une instance EC2 et la modification d'un groupe de sécurité, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise le démarrage ou l'arrêt d'une instance EC2 spécifique, ainsi que la modification d'un groupe de sécurité spécifique. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez celle de *italicized placeholder text* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSecurityGroupReferences",
        "ec2:DescribeStaleSecurityGroups"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "ec2:AuthorizeSecurityGroupEgress",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:RevokeSecurityGroupEgress",
        "ec2:RevokeSecurityGroupIngress",
        "ec2:StartInstances",
        "ec2:StopInstances"
      ],
      "Resource": [
        "arn:aws:ec2:*:*:instance/i-instance-id",
        "arn:aws:ec2:*:*:security-group/sg-security-group-id"
      ],
      "Effect": "Allow"
    }
  ]
```

```
}
```

Amazon EC2 : nécessite MFA (GetSessionToken) pour des opérations EC2 spécifiques

Cet exemple montre comment créer une politique basée sur l'identité qui permet un accès complet à toutes les opérations d' AWS API dans Amazon EC2. Cependant, il refuse explicitement l'accès aux opérations d'API `TerminateInstances` et `StopInstances` si l'utilisateur n'est pas authentifié à l'aide de l'authentification MFA. Pour le faire par programmation, l'utilisateur doit inclure des valeurs `SerialNumber` et `TokenCode` facultatives lors de l'appel de l'opération `GetSessionToken`. Cette opération renvoie les informations d'identification temporaires qui ont été authentifiées à l'aide de MFA. Pour en savoir plus GetSessionToken, consultezDemande d'informations d'identification temporaires pour les utilisateurs dans des environnements non fiables.

À quoi sert cette politique ?

- L'instruction `AllowAllActionsForEC2` autorise toutes les actions Amazon EC2.
- L'instruction `DenyStopAndTerminateWhenMFAIsNotPresent` rejette les actions `StopInstances` et `TerminateInstances` lorsque le contexte de MFA est manquant. Cela signifie que les actions sont refusées lorsque le contexte d'authentification multi-facteur est manquant (c'est-à-dire, quand l'authentification MFA n'a pas été utilisée). Un refus remplace l'autorisation.

> **Note**
>
> Le contrôle de la condition `MultiFactorAuthPresent` de l'instruction `Deny` ne doit pas être `{"Bool":{"aws:MultiFactorAuthPresent":false}}`, car cette clé n'est pas présente et ne peut pas être évaluée quand l'authentification MFA n'est pas utilisée. Utilisez donc plutôt le contrôle `BoolIfExists` pour voir si la clé est présente avant de vérifier la valeur. Pour de plus amples informations, veuillez consulter ... IfExists opérateurs conditionnelles.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllActionsForEC2",
            "Effect": "Allow",
            "Action": "ec2:*",
            "Resource": "*"
        },
        {
            "Sid": "DenyStopAndTerminateWhenMFAIsNotPresent",
            "Effect": "Deny",
            "Action": [
                "ec2:StopInstances",
                "ec2:TerminateInstances"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": false}
            }
        }
    ]
}
```

Amazon EC2 : limite la suspension des instances EC2 à une plage d'adresses IP

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui limite les instances EC2 en permettant l'action, mais en refusant explicitement l'accès lorsque la demande est externe à la plage d'adresses IP spécifiée. Cette politique est utile lorsque les adresses IP de votre entreprise se situent dans les plages d'adresses IP spécifiées. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *italicized placeholder text* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Si cette politique est utilisée en combinaison avec d'autres politiques autorisant l'`ec2:TerminateInstances`action (telles que la politique EC2 FullAccess AWS gérée par Amazon), l'accès est refusé. En effet, une instruction de refus explicite est prioritaire dans l'autorisation des instructions. Pour de plus amples informations, veuillez consulter Comment ? AWS la logique du code d'application évalue les demandes pour autoriser ou refuser l'accès.

> **Important**
>
> La clé de `aws:SourceIp` condition refuse l'accès à un Service AWS, tel que AWS CloudFormation, qui passe des appels en votre nom. Pour de plus amples informations sur l'utilisation de la clé de condition `aws:SourceIp`, veuillez consulter AWS clés de contexte de condition globale.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ec2:TerminateInstances"],
            "Resource": ["*"]
        },
        {
            "Effect": "Deny",
            "Action": ["ec2:TerminateInstances"],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "203.0.113.0/24"
                    ]
                }
            },
            "Resource": ["*"]
        }
    ]
}
```

IAM : Accès à l'API du simulateur de politique

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'utilisation de l'API du simulateur de politique pour les politiques attachées à un utilisateur, groupe d'utilisateurs ou rôle dans l' Compte AWS actuel. Cette politique autorise également d'accéder à la

simulation moins sensible des politiques transmises à l'API sous forme de chaînes. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForCustomPolicy",
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Pour autoriser un utilisateur à accéder à la console du simulateur de politiques afin de simuler des politiques associées à un utilisateur, à un groupe ou à un rôle dans le courant Compte AWS, consultezIAM : Accès à la console du simulateur de politiques.

IAM : Accès à la console du simulateur de politiques

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'utilisation de la console du simulateur de politique pour les politiques attachées à un utilisateur, groupe d'utilisateurs ou rôle dans l' Compte AWS actuel.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "access-analyzer:ValidatePolicy",
                "iam:GetContextKeysForCustomPolicy",
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:GetUser",
                "iam:GetUserPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListAttachedRolePolicies",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroups",
                "iam:ListGroupPolicies",
                "iam:ListGroupsForUser",
                "iam:ListPolicies",
                "iam:ListRolePolicies",
                "iam:ListRoles",
                "iam:ListUserPolicies",
                "iam:ListUsers",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

IAM : endosser des rôles qui ont une balise spécifique

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un utilisateur IAM à assumer des rôles avec la paire clé-valeur de balise `Project = ExampleCorpABC`. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Si un rôle avec cette balise existe dans le même compte que l'utilisateur, l'utilisateur peut endosser ce rôle. Si un rôle avec cette balise existe dans un compte autre que celui de l'utilisateur, il a besoin d'autorisations supplémentaires. La politique de confiance du rôle entre comptes doit également autoriser l'utilisateur ou tous les membres du compte de l'utilisateur à endosser le rôle. Pour de plus amples informations sur l'utilisation des rôles pour l'accès entre comptes, veuillez consulter Accès d'un utilisateur IAM à un autre Compte AWS que tu possèdes.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeTaggedRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:ResourceTag/Project": "ExampleCorpABC"}
            }
        }
    ]
}
```

IAM : autorise et refuse l'accès à plusieurs services par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un accès total à plusieurs services, ainsi qu'un accès autogéré limité dans IAM. Cela refuse également aux utilisateurs l'accès au compartiment des journaux Amazon S3 `logs` ou à l'instance Amazon EC2 `i-1234567890abcdef0` Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

> **Warning**
>
> Cette politique permet un accès total à toutes les actions et ressources dans plusieurs services. Cette politique doit être appliquée uniquement aux administrateurs de confiance.

Vous pouvez utiliser cette politique comme une limite des autorisations pour définir les autorisations maximales qu'une politique basée sur les identités peut accorder à un utilisateur IAM. Pour plus d'informations, veuillez consulter Délégation de responsabilité à d'autres utilisateurs à l'aide des limites d'autorisations. Lorsque la politique est utilisée comme une limite des autorisations pour un utilisateur, les instructions définissent les limites suivantes :

- L'instruction `AllowServices` autorise un accès complet au service AWS spécifié. Cela signifie que les nouvelles actions de l'utilisateur dans ces services sont uniquement limitées par les politiques d'autorisations qui lui sont attachées.
- L'instruction `AllowIAMConsoleForCredentials` autorise l'accès pour répertorier tous les utilisateurs IAM. Cet accès est nécessaire pour accéder à la page Utilisateurs dans la Console de gestion AWS. Il permet également d'afficher les exigences de mot de passe pour le compte, ce qui est nécessaire pour que l'utilisateur modifie son mot de passe.
- L'instruction `AllowManageOwnPasswordAndAccessKeys` autorise les utilisateurs à gérer uniquement leur propre mot de passe de console et les clés d'accès par programmation. C'est important, car si une autre politique accorde à un utilisateur l'accès complet à IAM, cet utilisateur peut ensuite modifier ses propres autorisations ou celles d'autres utilisateurs. Cette instruction évite ce genre de problème.
- L'instruction `DenyS3Logs` refuse explicitement l'accès au compartiment `logs`. Cette politique applique à l'entreprise des restrictions sur l'utilisateur.
- L'instruction `DenyEC2Production` refuse explicitement l'accès à l'instance `i-1234567890abcdef0`.

Cette politique n'autorise pas l'accès à d'autres services ou actions. Lorsque la politique est utilisée comme limite d'autorisations pour un utilisateur, même si d'autres politiques associées à l'utilisateur autorisent ces actions, AWS refuse la demande.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowServices",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*LoginProfile*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
```

```
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
        }
    ]
}
```

IAM : ajouter une balise spécifique à un utilisateur avec une balise spécifique

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'ajout de la clé de balise `Department` avec les valeurs de balise `Marketing`, `Development` ou `QualityAssurance` à un utilisateur IAM. Cet utilisateur doit déjà inclure la paire clé-valeur de la balise `JobFunction = Manager`. Vous pouvez utiliser cette politique pour exiger qu'un gestionnaire appartienne à un seul des trois départements. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

L'instruction `ListTagsForAllUsers` permet d'afficher les balises pour tous les utilisateurs de votre compte.

La première condition de l'instruction `TagManagerWithSpecificDepartment` utilise l'opérateur de condition `StringEquals`. La condition renvoie la valeur true si les deux parties de la condition sont true. L'utilisateur à baliser doit déjà posséder la balise `JobFunction=Manager`. La demande doit inclure la clé de balise `Department` avec l'une des valeurs de balise répertoriées.

La deuxième condition utilise l'opérateur de condition `ForAllValues:StringEquals`. La condition renvoie la valeur true si toutes les clés de balise dans la demande correspondent à la clé dans la politique. Cela signifie que la seule clé de balise dans la demande doit être `Department`. Pour plus d'informations sur l'utilisation de `ForAllValues`, consultez Définir les opérateurs pour les clés de contexte à valeurs multiples.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "ListTagsForAllUsers",
            "Effect": "Allow",
            "Action": [
                "iam:ListUserTags",
                "iam:ListUsers"
            ],
            "Resource": "*"
        },
        {
            "Sid": "TagManagerWithSpecificDepartment",
            "Effect": "Allow",
            "Action": "iam:TagUser",
            "Resource": "*",
            "Condition": {"StringEquals": {
                "iam:ResourceTag/JobFunction": "Manager",
                "aws:RequestTag/Department": [
                    "Marketing",
                    "Development",
                    "QualityAssurance"
                    ]
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "Department"}
            }
        }
    ]
}
```

IAM : ajouter une balise spécifique avec des valeurs spécifiques

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'ajout d'uniquement la clé de balise `CostCenter`, et soit de la valeur de balise `A-123`, soit de la valeur de balise `B-456` à n'importe quel utilisateur ou rôle IAM. Vous pouvez utiliser cette politique pour limiter le balisage à une clé de balise spécifique et de définir les valeurs de balise. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

L'instruction `ConsoleDisplay` permet d'afficher les balises pour tous les utilisateurs et rôles de votre compte.

La première condition de l'instruction `AddTag` utilise l'opérateur de condition `StringEquals`. La condition renvoie la valeur true si la demande inclut la clé de balise `CostCenter` avec l'une des valeurs de balise répertoriées.

La deuxième condition utilise l'opérateur de condition `ForAllValues:StringEquals`. La condition renvoie la valeur true si toutes les clés de balise dans la demande correspondent à la clé dans la politique. Cela signifie que la seule clé de balise dans la demande doit être `CostCenter`. Pour plus d'informations sur l'utilisation de `ForAllValues`, consultez Définir les opérateurs pour les clés de contexte à valeurs multiples.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ConsoleDisplay",
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:GetUser",
                "iam:ListRoles",
                "iam:ListRoleTags",
                "iam:ListUsers",
                "iam:ListUserTags"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AddTag",
            "Effect": "Allow",
            "Action": [
                "iam:TagUser",
                "iam:TagRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CostCenter": [
                        "A-123",
                        "B-456"
                    ]
```

```
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "CostCenter"}
            }
        }
    ]
}
```

IAM : créer des utilisateurs uniquement avec les balises spécifiques

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise la création d'utilisateurs IAM, mais uniquement avec l'une des clés de balise `Department` et `JobFunction`, ou les deux. La clé de balise `Department` doit avoir la valeur de balise `Development` ou `QualityAssurance`. La clé de balise `JobFunction` doit avoir la valeur de balise `Employee`. Vous pouvez utiliser cette politique pour exiger que les nouveaux utilisateurs aient une fonction et un service spécifiques. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

La première condition de l'instruction utilise l'opérateur de condition `StringEqualsIfExists`. Si une balise avec la clé `JobFunction` ou `Department` est présente dans la demande, la balise doit avoir la valeur spécifiée. Si aucune clé n'est présente, cette condition est évaluée comme true. Le seul moyen selon lequel la condition est évaluée en tant que false est si l'une des clés de condition spécifiées est présente dans la demande, mais possède une valeur différente de celles autorisées. Pour plus d'informations sur l'utilisation de `IfExists`, consultez ... IfExists opérateurs conditionnelles.

La deuxième condition utilise l'opérateur de condition `ForAllValues:StringEquals`. La condition renvoie la valeur true s'il existe une correspondance entre chacune des clés de balises spécifiées dans la demande et au moins une valeur dans la politique. Cela signifie que toutes les balises dans la demande doivent être dans cette liste. Toutefois, la demande peut inclure une seule des balises dans la liste. Par exemple, vous pouvez créer un utilisateur IAM avec la balise `Department=QualityAssurance` uniquement. Toutefois, vous ne pouvez pas créer un utilisateur IAM avec les balises `JobFunction=Employee` et `Project=core`. Pour plus d'informations sur l'utilisation de `ForAllValues`, consultez Définir les opérateurs pour les clés de contexte à valeurs multiples.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TagUsersWithOnlyTheseTags",
            "Effect": "Allow",
            "Action": [
                "iam:CreateUser",
                "iam:TagUser"
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "aws:RequestTag/Department": [
                        "Development",
                        "QualityAssurance"
                    ],
                    "aws:RequestTag/JobFunction": "Employee"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "Department",
                        "JobFunction"
                    ]
                }
            }
        }
    ]
}
```

IAM : générer et extraire des rapports sur les informations d'identification IAM

Cet exemple montre comment créer une politique basée sur l'identité qui permet aux utilisateurs de générer et de télécharger un rapport répertoriant tous les utilisateurs IAM dans leur Compte AWS Le rapport indique également le statut de plusieurs informations d'identification de l'utilisateur, y compris les mots de passe, les clés d'accès, les dispositifs MFA et les certificats de signature. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI

Pour de plus amples informations sur les rapports d'informations d'identification, veuillez consulter Générez des rapports d'identification pour votre Compte AWS.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateCredentialReport",
            "iam:GetCredentialReport"
        ],
        "Resource": "*"
    }
}
```

IAM : autorise la gestion des membres d'un groupe par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise la mise à jour de l'appartenance au groupe appelé `MarketingTeam`. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

À quoi sert cette politique ?

- L'instruction `ViewGroups` permet à l'utilisateur de répertorier tous les utilisateurs et groupes dans la Console de gestion AWS. Elle permet également à l'utilisateur de consulter des informations de base sur les utilisateurs dans le compte. Ces autorisations doivent se trouver dans leur propre instruction, car elles ne prennent pas en charge ou n'ont pas besoin de spécifier d'ARN de ressource particulier. À la place, les autorisations spécifient `"Resource" : "*"`.
- L'instruction `ViewEditThisGroup` permet à l'utilisateur d'afficher des informations sur le groupe `MarketingTeam`, ainsi que d'ajouter et de supprimer des utilisateurs de ce groupe.

Cette politique n'autorise pas l'utilisateur à afficher ou modifier les autorisations des utilisateurs ou du groupe `MarketingTeam`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ViewGroups",
            "Effect": "Allow",
            "Action": [
                "iam:ListGroups",
                "iam:ListUsers",
                "iam:GetUser",
                "iam:ListGroupsForUser"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ViewEditThisGroup",
            "Effect": "Allow",
            "Action": [
                "iam:AddUserToGroup",
                "iam:RemoveUserFromGroup",
                "iam:GetGroup"
            ],
            "Resource": "arn:aws:iam::*:group/MarketingTeam"
        }
    ]
}
```

IAM : gérer une balise spécifique

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'ajout et la suppression de la balise IAM avec la clé de balise `Department` à des entités IAM (utilisateurs et rôles). Cette politique ne limite pas la valeur de la balise `Department`. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`*

l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:TagUser",
            "iam:TagRole",
            "iam:UntagUser",
            "iam:UntagRole"

        ],
        "Resource": "*",
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys": "Department"}}
    }
}
```

IAM : transmettre un rôle IAM à un service spécifique AWS

Cet exemple montre comment créer une politique basée sur l'identité qui permet de transmettre n'importe quel rôle de service IAM au service Amazon. CloudWatch Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *italicized placeholder text* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Un rôle de service est un rôle IAM qui indique qu'un AWS service est le principal habilité à assumer le rôle. Cela permet au service d'endosser le rôle et d'accéder aux ressources dans d'autres services en votre nom. Pour permettre CloudWatch à Amazon d'assumer le rôle que vous lui transmettez, vous devez spécifier le principal du `cloudwatch.amazonaws.com` service comme principal dans la politique de confiance de votre rôle. Le principal de service est défini par le service. Pour connaître le principal de service d'un service, consultez la documentation de ce service. Pour certains services, consultez AWS services qui fonctionnent avec IAM et recherchez les services qui contiennent Oui dans la colonne Rôle lié à un service. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service. Recherchez `amazonaws.com` pour afficher le principal de service.

Pour en savoir plus sur la transmission d'un rôle de service à un service, consultez Accorder à un utilisateur l'autorisation de transmettre un rôle à un AWS web.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:PassedToService":
 "cloudwatch.amazonaws.com"}
            }
        }
    ]
}
```

IAM : autorise l'accès en lecture seule à la console IAM sans reporting

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise des utilisateurs IAM à effectuer toute action IAM dont le nom commence par la chaîne `Get` ou `List`. À mesure que les utilisateurs utilisent la console, celle-ci effectue des demandes à IAM pour répertorier les groupes, les utilisateurs, les rôles et les politiques, et pour générer des rapports sur ces ressources.

L'astérisque agit comme un caractère générique. Lorsque vous utilisez `iam:Get*` dans une politique, les autorisations obtenues incluent toutes les actions IAM commençant par `Get`, telles que `GetUser` et `GetRole`. Les caractères génériques sont utiles si de nouveaux types d'entités sont ajoutés à IAM à l'avenir. Dans ce cas, les autorisations accordées par la politique permettent automatiquement à l'utilisateur de répertorier et d'obtenir les détails relatifs à ces nouvelles entités.

Cette politique ne peut pas servir à générer des rapports ou des derniers détails de service consultés. Pour obtenir une politique différente l'autorisant, veuillez consulter IAM : autorise l'accès en lecture seule à la console IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    }
}
```

IAM : autorise l'accès en lecture seule à la console IAM

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise des utilisateurs IAM à effectuer toute action IAM dont le nom commence par la chaîne `Get`, `List` ou `Generate`. À mesure que les utilisateurs utilisent la console IAM, celle-ci effectue des demandes pour répertorier les groupes, les utilisateurs, les rôles et les politiques, et pour générer des rapports sur ces ressources.

L'astérisque agit comme un caractère générique. Lorsque vous utilisez `iam:Get*` dans une politique, les autorisations obtenues incluent toutes les actions IAM commençant par `Get`, telles que `GetUser` et `GetRole`. L'utilisation d'un caractère générique est avantageuse, en particulier si de nouveaux types d'entités sont ajoutés à IAM à l'avenir. Dans ce cas, les autorisations accordées par la politique permettent automatiquement à l'utilisateur de répertorier et d'obtenir les détails relatifs à ces nouvelles entités.

Utilisez cette politique pour un accès à la console incluant des autorisations pour générer des rapports ou les derniers détails du service consultés. Pour une politique différente n'autorisant pas la génération d'actions, veuillez consulter IAM : autorise l'accès en lecture seule à la console IAM sans reporting.

JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:Get*",
            "iam:List*",
            "iam:Generate*"
        ],
        "Resource": "*"
    }
}
```

IAM : autorise des utilisateurs spécifiques à gérer un groupe par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise des utilisateurs IAM spécifiques à gérer le groupe `AllUsers`. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

À quoi sert cette politique ?

- L'instruction `AllowAllUsersToListAllGroups` permet de répertorier tous les groupes. Ceci est nécessaire pour l'accès à la console. Cette autorisation doit se trouver dans sa propre instruction, car elle ne prend pas en charge un ARN de ressource. À la place, les autorisations spécifient `"Resource" : "*"`.
- L'instruction `AllowAllUsersToViewAndManageThisGroup` autorise toutes les actions du groupe pouvant être effectuées sur le type de ressource groupe. Elle n'autorise pas l'action `ListGroupsForUser`, qui peut être effectuée sur un type de ressource d'utilisateur et non pas un type de ressource de groupe. Pour de plus amples informations sur les types de ressources que vous pouvez spécifier pour une action IAM, veuillez consulter Actions, ressources et clés de condition pour Gestion des identités et des accès AWS.
- La `LimitGroupManagementAccessToSpecificUsers` déclaration refuse aux utilisateurs portant les noms spécifiés l'accès aux actions de groupe d'écriture et de gestion des autorisations. Lorsqu'un utilisateur spécifié dans la politique tente d'apporter des modifications au groupe, cette instruction ne permet pas de refuser la demande. Cette demande est autorisée par l'instruction `AllowAllUsersToViewAndManageThisGroup`. Si d'autres utilisateurs tentent d'effectuer

ces opérations, la demande est refusée. Vous pouvez afficher les actions IAM définies avec les niveaux d'accès Write (Écriture) ou Permissions management (Gestion des autorisations) lors de la création de cette politique dans la console IAM. Pour ce faire, passez de l'onglet JSON à l'onglet Visual editor (Éditeur visuel). Pour plus d'informations sur les niveaux d'accès, consultez la section Actions, ressources et clés de condition pour Gestion des identités et des accès AWS.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllUsersToListAllGroups",
            "Effect": "Allow",
            "Action": "iam:ListGroups",
            "Resource": "*"
        },
        {
            "Sid": "AllowAllUsersToViewAndManageThisGroup",
            "Effect": "Allow",
            "Action": "iam:*Group*",
            "Resource": "arn:aws:iam::*:group/AllUsers"
        },
        {
            "Sid": "LimitGroupManagementAccessToSpecificUsers",
            "Effect": "Deny",
            "Action": [
                "iam:AddUserToGroup",
                "iam:CreateGroup",
                "iam:RemoveUserFromGroup",
                "iam:DeleteGroup",
                "iam:AttachGroupPolicy",
                "iam:UpdateGroup",
                "iam:DetachGroupPolicy",
                "iam:DeleteGroupPolicy",
                "iam:PutGroupPolicy"
            ],
            "Resource": "arn:aws:iam::*:group/AllUsers",
            "Condition": {
                "StringNotEquals": {
                    "aws:username": [
```

```
                    "srodriguez",
                    "mjackson",
                    "adesai"
                ]
            }
        }
    ]
}
```

IAM : permet de définir les exigences de mot de passe du compte par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un utilisateur à afficher et mettre à jour ses exigences en matière de mot de passe de compte. Les exigences en matière de mot de passe spécifient les exigences de complexité et les intervalles de rotation obligatoires pour les mots de passe des membres du compte. Cette politique définit des autorisations pour l'accès à la console et par programmation.

Pour savoir comment définir la politique d'exigences de mot de passe de compte pour votre compte, consultez Définition d'une politique de mot de passe du compte pour les utilisateurs IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GetAccountPasswordPolicy",
            "iam:UpdateAccountPasswordPolicy"
        ],
        "Resource": "*"
    }
}
```

IAM : accès à l'API du simulateur de politique en fonction du chemin d'utilisateur

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'utilisation de l'API du simulateur de politique uniquement pour les utilisateurs ayant le chemin `Department/Development`. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
    ]
}
```

> **Note**
>
> Pour créer une politique qui autorise l'utilisation de la console du simulateur de politique uniquement pour les utilisateurs ayant un chemin `Department/Development`, consultez IAM : accès à la console du simulateur de politique en fonction du chemin d'utilisateur.

IAM : accès à la console du simulateur de politique en fonction du chemin d'utilisateur

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'utilisation de la console du simulateur de politique uniquement pour les utilisateurs ayant un chemin `Department/Development`. Pour utiliser cette politique, remplacez celle de *`italicized`*

placeholder text l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetPolicy",
                "iam:GetUserPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "iam:GetUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroupsForUser",
                "iam:ListUserPolicies",
                "iam:ListUsers"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
    ]
}
```

IAM : autorise les utilisateurs IAM à gérer eux-même un dispositif MFA

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise des utilisateurs IAM à autogérer leur périphérique à authentification multifactorielle (MFA). Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI

> **Note**
>
> Si un utilisateur IAM doté de cette politique n'est pas authentifié par MFA, cette politique refuse l'accès à toutes les AWS actions, à l'exception de celles nécessaires pour s'authentifier à l'aide de l'authentification MFA. Si vous ajoutez ces autorisations à un utilisateur connecté à AWS, il peut avoir besoin de se déconnecter puis de se reconnecter pour voir ces modifications.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToCreateVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowUserToManageTheirOwnMFA",
            "Effect": "Allow",
            "Action": [
                "iam:EnableMFADevice",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
```

```
        },
        {
            "Sid": "AllowUserToDeactivateTheirOwnMFAOnlyWhenUsingMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice"
            ],
            "Resource": [
                "arn:aws:iam::*:user/${aws:username}"
            ],
            "Condition": {
                "Bool": {
                    "aws:MultiFactorAuthPresent": "true"
                }
            }
        },
        {
            "Sid": "BlockMostAccessUnlessSignedInWithMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:ListMFADevices",
                "iam:ListUsers",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

IAM : Autoriser les utilisateurs IAM à changer leurs informations d'identification par programmation et dans la console

Cet exemple montre comment créer une politique basée sur l'identité qui permet aux utilisateurs IAM de mettre à jour leurs propres clés d'accès, certificats de signature, informations d'identification

spécifiques aux services et mots de passe. Cette politique définit des autorisations pour l'accès à la console et par programmation.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        }
    ]
}
```

Pour en savoir plus sur la façon dont les utilisateurs peuvent modifier leur propre mot de passe dans la console, consultez the section called "Modification par l'utilisateur IAM de son propre mot de passe".

IAM : Afficher les informations du dernier accès au service pour une politique AWS Organizations

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'affichage des dernières informations de service consultées pour une politique AWS Organizations spécifique. Cette politique autorise la récupération des données pour la politique de contrôle de

service (SCP) avec l'ID `p-policy123`. La personne qui génère et consulte le rapport doit être authentifiée à l'aide AWS Organizations des informations d'identification du compte de gestion. Cette politique permet au demandeur de récupérer les données de n'importe quelle AWS Organizations entité de son organisation. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Pour des informations importantes sur les dernières informations consultées, y compris les autorisations requises, la résolution de problèmes et les régions prises en charge, veuillez consulter Affiner les autorisations en AWS utilisant les dernières informations consultées.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOrgsReadOnlyAndIamGetReport",
            "Effect": "Allow",
            "Action": [
                "iam:GetOrganizationsAccessReport",
                "organizations:Describe*",
                "organizations:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowGenerateReportOnlyForThePolicy",
            "Effect": "Allow",
            "Action": "iam:GenerateOrganizationsAccessReport",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:OrganizationsPolicyId": "p-policy123"}
            }
        }
    ]
}
```

IAM : limite les politiques gérées pouvant être appliquées à un utilisateur, groupe ou rôle IAM

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui limite les politiques AWS gérées par le client et qui peuvent être appliquées à un utilisateur, un groupe ou un rôle IAM. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *italicized placeholder text* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachUserPolicy",
            "iam:DetachUserPolicy"
        ],
        "Resource": "*",
        "Condition": {
            "ArnEquals": {
                "iam:PolicyARN": [
                    "arn:aws:iam::*:policy/policy-name-1",
                    "arn:aws:iam::*:policy/policy-name-2"
                ]
            }
        }
    }
}
```

AWS: Refusez l'accès aux ressources extérieures à votre compte, à l'exception des AWS politiques IAM gérées

L'utilisation de `aws:ResourceAccount` dans vos politiques basées sur l'identité peut avoir un impact sur la capacité de l'utilisateur ou du rôle à utiliser certains services qui nécessitent une interaction avec des ressources dans des comptes appartenant à un service.

Vous pouvez créer une politique avec une exception pour autoriser les politiques IAM AWS gérées. Un compte géré par un service en dehors de vos propres politiques AWS Organizations IAM gérées. Il existe quatre actions IAM qui répertorient et récupèrent les AWS politiques gérées. Utilisez ces actions dans l'élément `NotAction` de la déclaration `AllowAccessToResourcesInSpecificAccountsAndSpecificService1` dans la politique.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessToResourcesInSpecificAccountsAndSpecificService1",
      "Effect": "Deny",
      "NotAction": [
        "iam:GetPolicy",
        "iam:GetPolicyVersion",
        "iam:ListEntitiesForPolicy",
        "iam:ListPolicies"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    }
  ]
}
```

AWS Lambda : autorise une fonction Lambda à accéder à une table Amazon DynamoDB

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès en lecture et en écriture à une table Amazon DynamoDB spécifique. La politique permet également d'écrire des fichiers CloudWatch journaux dans Logs. Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres

informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Pour utiliser cette politique, attachez la politique à un rôle de service Lambda. Un rôle de service est un rôle que vous créez dans votre compte pour autoriser un service à effectuer des actions en votre nom. Ce rôle de service doit figurer AWS Lambda en tant que principal dans la politique de confiance. Pour en savoir plus sur l'utilisation de cette politique, consultez Comment créer une politique AWS IAM pour accorder l' AWS Lambda accès à une table AWS Amazon DynamoDB dans le blog sur la sécurité.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ReadWriteTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb:BatchGetItem",
                "dynamodb:GetItem",
                "dynamodb:Query",
                "dynamodb:Scan",
                "dynamodb:BatchWriteItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable"
        },
        {
            "Sid": "GetStreamRecords",
            "Effect": "Allow",
            "Action": "dynamodb:GetRecords",
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable/stream/* "
        },
        {
            "Sid": "WriteLogStreamsAndGroups",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
```

```
            "Resource": "*"
        },
        {
            "Sid": "CreateLogGroup",
            "Effect": "Allow",
            "Action": "logs:CreateLogGroup",
            "Resource": "*"
        }
    ]
}
```

Amazon RDS : autorise l'accès complet à la base de données RDS dans une région spécifique

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès total à la base de données RDS dans une région spécifique. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *italicized placeholder text* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "rds:*",
            "Resource": ["arn:aws:rds:us-east-1:*:*"]
        },
        {
            "Effect": "Allow",
            "Action": ["rds:Describe*"],
            "Resource": ["*"]
        }
    ]
}
```

Amazon RDS : autorise la restauration des bases de données RDS, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise la restauration de bases de données RDS. Cette politique définit des autorisations pour l'accès à la console et par programmation.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "rds:CreateDBParameterGroup",
                "rds:CreateDBSnapshot",
                "rds:DeleteDBSnapshot",
                "rds:Describe*",
                "rds:DownloadDBLogFilePortion",
                "rds:List*",
                "rds:ModifyDBInstance",
                "rds:ModifyDBParameterGroup",
                "rds:ModifyOptionGroup",
                "rds:RebootDBInstance",
                "rds:RestoreDBInstanceFromDBSnapshot",
                "rds:RestoreDBInstanceToPointInTime"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon RDS : autorise aux propriétaires de balise l'accès complet aux ressource RDS qu'ils ont balisées

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui offre aux propriétaires de balises un accès total aux ressources RDS qu'ils ont balisées. Cette politique

accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "rds:Describe*",
                "rds:List*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "rds:DeleteDBInstance",
                "rds:RebootDBInstance",
                "rds:ModifyDBInstance"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:db-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyOptionGroup",
                "rds:DeleteOptionGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:og-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyDBParameterGroup",
                "rds:ResetDBParameterGroup"
```

```
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:pg-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:AuthorizeDBSecurityGroupIngress",
                "rds:RevokeDBSecurityGroupIngress",
                "rds:DeleteDBSecurityGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:secgrp-tag/Owner":
"${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:DeleteDBSnapshot",
                "rds:RestoreDBInstanceFromDBSnapshot"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:snapshot-tag/Owner":
"${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyDBSubnetGroup",
                "rds:DeleteDBSubnetGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:subgrp-tag/Owner":
"${aws:username}"}
            }
        },
```

```
        {
            "Action": [
                "rds:ModifyEventSubscription",
                "rds:AddSourceIdentifierToSubscription",
                "rds:RemoveSourceIdentifierFromSubscription",
                "rds:DeleteEventSubscription"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:es-tag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

Amazon S3 : permet aux utilisateurs Amazon Cognito d'accéder aux objets dans leur compartiment

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise des utilisateurs Amazon Cognito à accéder à des objets dans un compartiment Amazon S3 spécifique. Cette politique permet d'accéder uniquement aux objets comportant le mot `cognito`, le nom de l'application, ainsi que l'ID du principal fédéré, représenté par la variable ${cognito-identity.amazonaws.com:sub}. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

> **Note**
>
> La valeur « sub » utilisée dans la clé d'objet n'est pas la sous-valeur de l'utilisateur dans le groupe d'utilisateurs, c'est l'ID d'identité associé à l'utilisateur dans le groupe d'identités.

JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement": [
    {
      "Sid": "ListYourObjects",
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": [
        "arn:aws:s3:::bucket-name"
      ],
      "Condition": {
        "StringLike": {
          "s3:prefix": [
            "cognito/application-name/${cognito-identity.amazonaws.com:sub}/*"
          ]
        }
      }
    },
    {
      "Sid": "ReadWriteDeleteYourObjects",
      "Effect": "Allow",
      "Action": [
        "s3:DeleteObject",
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::bucket-name/cognito/application-name/${cognito-
identity.amazonaws.com:sub}/*"
      ]
    }
  ]
}
```

Amazon Cognito assure l'authentification, l'autorisation et la gestion des utilisateurs pour vos applications web et mobiles. Vos utilisateurs peuvent se connecter directement avec un nom d'utilisateur et un mot de passe, ou via un tiers tels que Facebook, Amazon ou Google.

Les deux principaux composants d'Amazon Cognito sont les groupes d'utilisateurs et les groupes d'identités. Les groupes d'utilisateurs sont des répertoires d'utilisateurs qui fournissent des options d'inscription et de connexion pour les utilisateurs de votre application. Les groupes d'identités vous permettent d'accorder à vos utilisateurs l'accès à d'autres AWS services. Vous pouvez utiliser des groupes d'identités et des groupes d'utilisateurs séparément ou conjointement.

Pour plus d'informations sur Amazon Cognito, consultez le Guide de l'utilisateur Amazon Cognito.

Amazon S3 : autorise les utilisateurs fédérés à accéder à leur répertoire de base Amazon S3, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise des principaux fédérés à accéder à leur propre objet de compartiment de répertoire de base dans S3. Le répertoire de base est un compartiment qui inclut un dossier `home` et des dossiers pour les principaux fédérés individuels. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

La variable `${aws:userid}` de cette politique se résout à `role-id:specified-name`. La partie `role-id` de l'ID du principal fédéré est un ID unique attribué au rôle de du principal fédéré au moment de la création. Pour de plus amples informations, veuillez consulter Identifiants uniques. `specified-name`Il s'agit du RoleSessionName paramètre transmis à la `AssumeRoleWithWebIdentity` demande lorsque le principal fédéré a assumé son rôle.

Vous pouvez afficher l'ID du rôle à l'aide de la AWS CLI commande`aws iam get-role --role-name` *`specified-name`*. Par exemple, imaginons que vous spécifiez le nom convivial `John` et que l'CLI renvoie l'ID de rôle `AROAXXT2NJT7D3SIQN7Z6`. Dans ce cas, l'ID du principal fédéré est `AROAXXT2NJT7D3SIQN7Z6:John`. Cette politique permet ensuite au principal fédéré John d'accéder au compartiment Amazon S3 avec le préfixe `AROAXXT2NJT7D3SIQN7Z6:John`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
```

```
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:userid}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:userid}",
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:userid}/*"
            ]
        }
    ]
}
```

Amazon S3 : accès au compartiment S3, mais compartiment de production refusé sans MFA récente

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un administrateur Amazon S3 à accéder à n'importe quel compartiment, notamment pour la mise à jour, l'ajout et la suppression d'objets. Cependant, il refuse explicitement l'accès au compartiment `amzn-s3-demo-bucket-production` si l'utilisateur ne s'est pas connecté via Multi-Factor Authentication (MFA) au cours des trente dernières minutes. Cette politique accorde les autorisations nécessaires pour effectuer cette action dans la console ou par programmation à l'aide de l'API AWS CLI or AWS . Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de

politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Cette politique n'autorise jamais un accès par programme au compartiment `amzn-s3-demo-bucket` à l'aide des clés d'accès de l'utilisateur à long terme. Ceci est possible en utilisant la clé de condition `aws:MultiFactorAuthAge` avec l'opérateur de condition `NumericGreaterThanIfExists`. Cette condition de politique renvoie `true` si MFA n'est pas présent ou si l'âge de la MFA est supérieur à 30 minutes. Dans ces situations, l'accès est refusé. Pour accéder au `amzn-s3-demo-bucket-production` compartiment par programmation, l'administrateur S3 doit utiliser des informations d'identification temporaires qui ont été générées au cours des 30 dernières minutes à l'aide de l'opération GetSessionTokenAPI.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListAllS3Buckets",
            "Effect": "Allow",
            "Action": ["s3:ListAllMyBuckets"],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowBucketLevelActions",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowBucketObjectActions",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:DeleteObject"
```

```
            ],
            "Resource": "arn:aws:s3:::*/*"
        },
        {
            "Sid": "RequireMFAForProductionBucket",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket-production/*",
                "arn:aws:s3:::amzn-s3-demo-bucket-production"
            ],
            "Condition": {
                "NumericGreaterThanIfExists": {"aws:MultiFactorAuthAge": "1800"}
            }
        }
    ]
}
```

Amazon S3 : autorise les utilisateurs IAM à accéder à leur répertoire de base S3, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise des utilisateurs IAM à accéder à leur propre objet de compartiment de répertoire de base dans S3. Le répertoire d'accueil est un compartiment qui inclut un dossier home et des dossiers pour les utilisateurs individuels. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez celle de *italicized placeholder text* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Cette politique ne fonctionne pas lors de l'utilisation de rôles IAM, car la variable `aws:username` n'est pas disponible lorsque vous utilisez des rôles IAM. Pour plus d'informations sur les valeurs de clé principale, consultez la section Valeurs de la clé du principal.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
```

```
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:username}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}",
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}/*"
            ]
        }
    ]
}
```

Amazon S3 : restreindre la gestion à un compartiment S3 spécifique

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui limite la gestion d'un compartiment Amazon S3 à ce compartiment spécifique. Cette politique accorde

l'autorisation d'effectuer toutes les actions Amazon S3, mais refuse l'accès à tous les Service AWS , à l'exception d'Amazon S3. Consultez l'exemple suivant. Selon cette politique, vous ne pouvez accéder qu'aux actions Amazon S3 que vous pouvez effectuer sur un compartiment S3 ou une ressource objet S3. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *italicized placeholder text* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Si cette politique est utilisée en combinaison avec d'autres politiques (telles que les politiques EC2 FullAccess AWS gérées par Amazon ou FullAccess Amazon S3) qui autorisent les actions interdites par cette politique, l'accès est refusé. En effet, une instruction de refus explicite est prioritaire dans l'autorisation des instructions. Pour de plus amples informations, veuillez consulter Comment ? AWS la logique du code d'application évalue les demandes pour autoriser ou refuser l'accès.

> **Warning**
>
> `NotAction` et `NotResource` sont des éléments de politique avancée qui doivent être utilisés avec précaution. Cette politique refuse l'accès à chaque service AWS , à l'exception d'Amazon S3. Si vous attachez cette politique à un utilisateur, toutes les autres politiques qui accordent des autorisations aux autres services sont ignorées et l'accès est refusé.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        },
        {
            "Effect": "Deny",
            "NotAction": "s3:*",
```

```
            "NotResource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        }
    ]
}
```

Octroi d'un accès en lecture et en écriture aux objets du compartiment Amazon S3

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès en `Read` et en `Write` aux objets d'un compartiment Amazon S3 spécifique. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

L'action `s3:*Object` utilise un caractère générique dans le nom de l'action. L'instruction `AllObjectActions` autorise les actions `GetObject`, `DeleteObject`, `PutObject` et toute autre action Amazon S3 qui se termine par le mot « Object ».

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": ["s3:ListBucket"],
            "Resource": ["arn:aws:s3:::bucket-name"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::bucket-name/*"]
        }
    ]
}
```

> **Note**
>
> Pour autoriser l'accès en `Read` et en `Write` aux objets d'un compartiment Amazon S3 spécifique et inclure également des autorisations supplémentaires pour l'accès à la console, consultez Amazon S3 : autorise l'accès en lecture et en écriture aux objets d'un compartiment S3, par programmation et dans la console.

Amazon S3 : autorise l'accès en lecture et en écriture aux objets d'un compartiment S3, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès en `Read` et en `Write` aux objets d'un compartiment S3 spécifique. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

L'action `s3:*Object` utilise un caractère générique dans le nom de l'action. L'instruction `AllObjectActions` autorise les actions `GetObject`, `DeleteObject`, `PutObject` et toute autre action Amazon S3 qui se termine par le mot « Object ».

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
```

```
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/*"]
        }
    ]
}
```

Gestion des politiques IAM

IAM vous fournit les outils pour créer et gérer tous types de politiques IAM (politiques gérées et politiques en ligne). Pour ajouter des autorisations à une identité IAM (utilisateur, groupe ou rôle IAM) vous créez une politique , vous la validez, puis vous l'attachez à l'identité. Vous pouvez attacher plusieurs politiques à une identité, et chaque politique peut contenir plusieurs autorisations.

Rubriques

- Ressources supplémentaires
- Définition d'autorisations IAM personnalisées avec des politiques gérées par le client
- Validation de politique IAM
- Test de politique IAM avec le simulateur de politiques IAM
- Ajout et suppression d'autorisations basées sur l'identité IAM
- Gestion des versions des politiques IAM
- Modification de politiques IAM
- Suppression de politiques IAM
- Affiner les autorisations en AWS utilisant les dernières informations consultées

Ressources supplémentaires

Les ressources suivantes peuvent vous aider à en savoir plus sur AWS les politiques.

- Pour plus d'informations sur les différents types de politiques IAM, consultez Politiques et autorisations dans Gestion des identités et des accès AWS.
- Pour obtenir des informations générales sur l'utilisation des politiques dans IAM, consultez Gestion des accès pour AWS resources.
- Pour plus d'informations sur la façon d'utiliser l'analyseur d'accès IAM pour générer une politique IAM basée sur une activité d'accès pour une entité, consultez Génération d'une politique de l'analyseur d'accès IAM.
- Pour de plus amples informations sur l'évaluation des autorisations lorsque plusieurs politiques sont en vigueur pour une identité IAM donnée, veuillez consulter Logique d'évaluation de politiques.
- Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

Définition d'autorisations IAM personnalisées avec des politiques gérées par le client

Les politiques définissent les autorisations pour les identités ou les ressources dans AWS. Vous pouvez créer des politiques gérées par le client dans IAM à l'aide de l' AWS API Console de gestion AWS AWS CLI, ou. Les politiques gérées par le client sont des politiques autonomes que vous gérez dans votre propre Compte AWS. Vous pouvez ensuite associer les politiques aux identités (utilisateurs, groupes et rôles) de votre Compte AWS.

Une politique basée sur l'identité est une politique attachée à une identité dans IAM. Les politiques basées sur l'identité peuvent inclure des politiques AWS gérées, des politiques gérées par le client et des politiques intégrées. AWS les politiques gérées sont créées et gérées par AWS, et vous pouvez les utiliser mais pas les gérer. Une politique en ligne peut être créée et intégrée directement à un groupes d'utilisateurs, un utilisateur ou un rôle IAM. Les politiques en ligne ne peuvent pas être réutilisées avec d'autres identités ni être gérées en dehors de l'identité où elles existent. Pour de plus amples informations, veuillez consulter Ajout et suppression d'autorisations basées sur l'identité IAM.

Il est généralement préférable d'utiliser des politiques gérées par le client plutôt que des politiques intégrées ou des politiques AWS gérées. AWS les politiques gérées fournissent généralement des autorisations administratives ou en lecture seule étendues. Pour plus de sécurité, octroyez le

moindre privilège, c'est-à-dire n'accordez que les autorisations nécessaires à l'exécution de tâches spécifiques.

Lorsque vous créez ou modifiez des politiques IAM, vous AWS pouvez effectuer automatiquement la validation des politiques pour vous aider à créer une politique efficace avec le moindre privilège à l'esprit. Dans le Console de gestion AWS, IAM identifie les erreurs de syntaxe JSON, tandis qu'IAM Access Analyzer fournit des vérifications de politique supplémentaires avec des recommandations pour vous aider à affiner davantage vos politiques. Pour en savoir plus sur la validation de politiques, veuillez consulter Validation de politique IAM. Pour en savoir plus sur les vérifications des politiques IAM Access Analyzer et les recommandations exploitables, veuillez consulter Validation de politique IAM Access Analyzer.

Vous pouvez utiliser l' AWS API Console de gestion AWS AWS CLI, ou pour créer des politiques gérées par le client dans IAM. Pour plus d'informations sur l'utilisation de CloudFormation modèles pour ajouter ou mettre à jour des politiques, consultez la référence aux types de Gestion des identités et des accès AWS ressources dans le Guide de CloudFormation l'utilisateur.

Rubriques

- Création de politiques IAM (console)
- Création de politiques IAM (AWS CLI)
- Création de politiques IAM (AWS API)

Création de politiques IAM (console)

Une politique est une entité qui, lorsqu'elle est attachée une identité ou à une ressource, définit les autorisations de cette dernière. Vous pouvez utiliser le Console de gestion AWS pour créer des politiques gérées par le client dans IAM. Les politiques gérées par le client sont des politiques autonomes que vous gérez dans votre propre Compte AWS. Vous pouvez ensuite associer les politiques aux identités (utilisateurs, groupes et rôles) de votre Compte AWS.

Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

Rubriques

- Création de politiques IAM
- Création de politiques à l'aide de l'éditeur JSON
- Création de politiques avec l'éditeur visuel

- Importation de politiques gérées existantes

Création de politiques IAM

Vous pouvez créer une politique gérée par le client en Console de gestion AWS utilisant l'une des méthodes suivantes :

- JSON — coller et personnaliser un exemple de politique basée sur l'identité publié.
- Visual editor (Éditeur visuel) — construire intégralement une nouvelle politique dans l'éditeur visuel. Si vous utilisez l'éditeur visuel, vous n'avez pas besoin de comprendre la syntaxe JSON.
- Import (Importation) — importer et personnaliser une politique gérée depuis votre compte. Vous pouvez importer une politique AWS gérée ou une politique gérée par le client que vous avez créée précédemment.

Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

Création de politiques à l'aide de l'éditeur JSON

Vous pouvez taper ou coller des politiques dans JSON à l'aide de l'option JSON. Cette méthode est utile pour copier un exemple de politique à utiliser dans votre compte. Ou vous pouvez composer votre propre document de politique JSON dans l'éditeur JSON. Vous pouvez également utiliser l'option JSON pour basculer entre l'éditeur visuel et JSON afin de comparer les vues.

Lorsque vous créez ou modifiez une politique dans l'éditeur JSON, IAM effectue une validation de politique pour vous aider à créer une politique efficace. IAM identifie les erreurs de syntaxe JSON, tandis qu'IAM Access Analyzer fournit des vérifications de politique supplémentaires avec des recommandations pratiques pour vous aider à affiner la politique.

Un document de politique JSON est composé d'une ou plusieurs instructions. Chaque instruction doit contenir toutes les actions qui partagent le même effet (`Allow` ou `Deny`) et qui prennent en charge les mêmes ressources et conditions. Si une action nécessite que toutes les ressources soient spécifiées (`"*"`) et qu'une autre action prend en charge l'Amazon Resource Name (ARN) d'une ressource spécifique, elles doivent se trouver dans deux instructions JSON distinctes. Pour plus d'informations sur les formats ARN, consultez Amazon Resource Name (ARN) dans le guide Références générales AWS . Pour obtenir des informations d'ordre général sur les politiques IAM, consultez Politiques et autorisations dans Gestion des identités et des accès AWS. Pour en savoir plus sur le langage de politique IAM, consultez Référence de politique JSON IAM.

Pour utiliser l'éditeur de politique JSON afin de créer une politique

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de gauche, choisissez Politiques.
3. Choisissez Create Policy (Créer une politique).
4. Dans la section Éditeur de politiques, choisissez l'option JSON.
5. Composez ou collez un document de politique JSON. Pour plus d'informations sur le langage de politique IAM, consultez Référence de politique JSON IAM.
6. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant.

   > **Note**
   >
   > Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour de plus amples informations, veuillez consulter Restructuration de politique.

7. (Facultatif) Lorsque vous créez ou modifiez une politique dans le Console de gestion AWS, vous pouvez générer un modèle de stratégie JSON ou YAML que vous pouvez utiliser dans les CloudFormation modèles.

   Pour ce faire, dans l'éditeur de politiques, sélectionnez Actions, puis sélectionnez Générer CloudFormation un modèle. Pour en savoir plus, CloudFormation consultez la référence Gestion des identités et des accès AWS aux types de ressources dans le guide de AWS CloudFormation l'utilisateur.
8. Lorsque vous avez fini d'ajouter des autorisations à la politique, choisissez Suivant.
9. Sur la page Vérifier et créer, tapez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vérifiez les Autorisations définies dans cette politique pour voir les autorisations accordées par votre politique.
10. (Facultatif) Ajoutez des métadonnées à la politique en associant les balises sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Tags pour Gestion des identités et des accès AWS resources.
11. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

Une fois que vous avez créé une politique, vous pouvez l'attacher à vos utilisateurs, groupes ou rôles. Pour plus d’informations, veuillez consulter Ajout et suppression d'autorisations basées sur l'identité IAM.

Création de politiques avec l'éditeur visuel

L'éditeur visuel de la console IAM vous guide au cours de la création d'une politique sans que vous ayez besoin d'écrire une syntaxe JSON. Pour voir un exemple d'utilisation de l'éditeur visuel pour la création d'un politique, consultez the section called “Contrôle de l'accès aux identités”.

Pour utiliser l'éditeur visuel afin de créer une politique

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de gauche, choisissez Politiques.
3. Choisissez Create Policy (Créer une politique).
4. Dans la section Éditeur de politiques, recherchez la section Sélectionner un service, puis choisissez un AWS service. Vous pouvez utiliser le menu Filtre ou la zone de recherche en haut de l'écran pour limiter les résultats dans la liste des services. Vous pouvez choisir un seul service par bloc d'autorisation de l'éditeur visuel. Pour accorder l'accès à plusieurs services, ajoutez de multiples blocs d'autorisation en sélectionnant Ajouter d'autres autorisations.
5. Dans Actions autorisées, choisissez les actions à ajouter à la politique. Vous pouvez choisir des actions de différentes manières :

   - Cochez la case à cocher pour toutes les actions.
   - Choisissez Ajouter des actions pour saisir le nom d'une action spécifique. Vous pouvez utiliser des caractères génériques (*) pour spécifier plusieurs actions.
   - Sélectionnez l'un des groupes de niveau Accès pour choisir toutes les actions pour le niveau d'accès (par exemple, Lecture, Écriture ou Liste).
   - Développez chacun des groupes de Niveaux d'accès pour choisir des actions individuelles.

   Par défaut, la politique que vous créez autorise les actions que vous choisissez. Pour refuser les actions choisies, sélectionnez Switch to deny permissions (Basculer vers le refus des autorisations). Le comportement par défaut d'IAM étant le refus, nous vous recommandons comme bonne pratique de sécurité de n'autoriser un utilisateur à accéder qu'aux actions et aux ressources nécessaires. Vous devez créer une instruction JSON pour refuser des autorisations seulement si vous souhaitez remplacer une autorisation séparément autorisée par une autre

instruction ou politique. Nous vous recommandons de limiter le nombre de refus d'autorisation au minimum, car ils peuvent rendre la résolution des problèmes d'autorisation plus complexe.

6. Pour Ressources, si le service et les actions que vous avez sélectionnés lors des étapes précédentes ne prennent pas en charge le choix de ressources spécifiques, toutes les ressources sont autorisées et vous ne pouvez pas modifier cette section.

   Si vous avez choisi une ou plusieurs actions qui prennent en charge les autorisations de niveau ressource, l'éditeur visuel affiche la liste de ces ressources. Vous pouvez alors développer Ressources pour spécifier les ressources de votre politique.

   Vous pouvez spécifier des ressources de la manière suivante :

   - Choisissez Ajouter ARNs pour spécifier les ressources en fonction de leur Amazon Resource Names (ARN). Vous pouvez utiliser l'éditeur visuel d'ARN ou la liste ARNs manuellement. Pour de plus amples informations sur la syntaxe ARN, veuillez consulter Amazon Resource Name (ARN) dans le Guide Références générales AWS . Pour plus d'informations sur l'utilisation ARNs dans l'`Resource`élément d'une politique, consultezÉléments de la politique JSON IAM : Ressource.
   - Choisissez Toute ressource dans ce compte en regard d'une ressource pour accorder des autorisations à toutes les ressources de ce type.
   - Choisissez Toutes pour choisir toutes les ressources pour le service.

7. (Facultatif) Choisissez Demander des conditions – facultatif pour ajouter des conditions à la politique que vous créez. Des conditions limitent l'effet d'une instruction de politique JSON. Par exemple, vous pouvez spécifier qu'un utilisateur est autorisé à effectuer des actions sur les ressources uniquement si la demande de cet utilisateur se produit au cours d'une période spécifiée. Vous pouvez également faire appel à des conditions couramment utilisées pour limiter le fait qu'un utilisateur doive être authentifié à l'aide d'un dispositif MFA (authentification multifacteur). Ou vous pouvez exiger que la demande provienne d'une certaine plage d'adresses IP. Pour obtenir la liste de toutes les clés contextuelles que vous pouvez utiliser dans une condition de politique, consultez la section Actions, ressources et clés de condition pour les AWS services dans la référence d'autorisation de service.

   Vous pouvez choisir des conditions de différentes manières :

   - Utilisez les cases à cocher pour sélectionner les conditions couramment utilisées.
   - Choisissez Ajouter une autre condition pour spécifier d'autres conditions. Choisissez la Clé de condition, le Qualificateur et l'Opérateur de la condition, puis saisissez une Valeur. Pour

ajouter plusieurs valeurs, choisissez Ajouter. Vous pouvez considérer que les valeurs sont connectées par un opérateur logique « OU ». Lorsque vous avez fini, choisissez Ajouter une condition.

Pour ajouter plusieurs conditions, choisissez de nouveau Ajouter une autre condition. Répétez l'opération si nécessaire. Chaque condition s'applique uniquement à ce bloc d'autorisation de l'éditeur visuel. Toutes les conditions doivent être remplies pour que le bloc d'autorisation soit considérée comme réussi. En d'autres termes, considérez les conditions comme étant connectées par un opérateur logique « ET ».

Pour plus d'informations sur l'élément Condition, consultez Éléments de la politique JSON IAM : Condition dans le document Référence de politique JSON IAM.

8. Pour ajouter d'autres blocs d'autorisation, choisissez Ajouter d'autres autorisations. Pour chaque bloc, répétez les étapes 2 à 5.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour de plus amples informations, veuillez consulter Restructuration de politique.

9. (Facultatif) Lorsque vous créez ou modifiez une politique dans le Console de gestion AWS, vous pouvez générer un modèle de stratégie JSON ou YAML que vous pouvez utiliser dans les CloudFormation modèles.

   Pour ce faire, dans l'éditeur de politiques, sélectionnez Actions, puis sélectionnez Générer CloudFormation un modèle. Pour en savoir plus, CloudFormation consultez la référence Gestion des identités et des accès AWS aux types de ressources dans le guide de AWS CloudFormation l'utilisateur.

10. Lorsque vous avez fini d'ajouter des autorisations à la politique, choisissez Suivant.

11. Sur la page Vérifier et créer, tapez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vérifiez les Autorisations définies dans cette politique pour vous assurer que vous les avez accordées comme prévu.

12. (Facultatif) Ajoutez des métadonnées à la politique en associant les balises sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Tags pour Gestion des identités et des accès AWS resources.
13. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

Une fois que vous avez créé une politique, vous pouvez l'attacher à vos utilisateurs, groupes ou rôles. Pour plus d'informations, veuillez consulter Ajout et suppression d'autorisations basées sur l'identité IAM.

Importation de politiques gérées existantes

Une manière facile de créer une nouvelle politique consiste à importer dans votre compte une politique gérée existante qui possède au moins certains des autorisations dont vous avez besoin. Vous pouvez ensuite personnaliser cette politique pour qu'elle corresponde à vos nouveaux besoins.

Vous ne pouvez pas importer une politique en ligne. Pour en savoir plus sur la différence entre les politiques gérées et les politiques en ligne, consultez Politiques gérées et politiques en ligne.

Pour importer une politique gérée existante dans l'éditeur visuel

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de gauche, choisissez Politiques.
3. Choisissez Create Policy (Créer une politique).
4. Dans l'Éditeur de politiques, choisissez Visuel, puis, sur le côté droit de la page, choisissez Actions, puis Importer une politique.
5. Dans la fenêtre Importer une politique, choisissez les politiques gérées qui correspondent le mieux à celle que vous voulez inclure dans votre nouvelle politique. Vous pouvez utiliser la zone de recherche en haut de la page pour limiter les résultats dans la liste des politiques.
6. Choisissez Importer une politique.

   Les politiques importées sont ajoutées dans de nouveaux blocs d'autorisation au bas de votre politique.
7. Utilisez Éditeur visuel ou choisissez JSON pour personnaliser votre politique. Ensuite, sélectionnez Suivant.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour de plus amples informations, veuillez consulter Restructuration de politique.

8. Sur la page Vérifier et créer, tapez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vous ne pourrez plus modifier ces paramètres ultérieurement. Vérifiez les Autorisations définies dans cette politique, puis choisissez Créer une politique pour enregistrer votre travail.

Pour importer une politique gérée existante dans l'éditeur JSON

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de gauche, choisissez Politiques.
3. Choisissez Create Policy (Créer une politique).
4. Dans la section Éditeur de politiques, choisissez l'option JSON, puis, sur le côté droit de la page, choisissez Actions, puis Importer une politique.
5. Dans la fenêtre Importer une politique, choisissez les politiques gérées qui correspondent le mieux à celle que vous voulez inclure dans votre nouvelle politique. Vous pouvez utiliser la zone de recherche en haut de la page pour limiter les résultats dans la liste des politiques.
6. Choisissez Importer une politique.

   Les instructions des politiques importées sont ajoutées au bas de votre politique JSON.
7. Personnalisez votre politique dans JSON. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant. Personnalisez votre politique dans JSON ou sélectionnez Visual editor (Éditeur visuel). Ensuite, sélectionnez Suivant.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur

visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour de plus amples informations, veuillez consulter Restructuration de politique.

8. Sur la page Vérifier et créer, tapez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vous ne pourrez plus modifier ces champs ultérieurement. Vérifiez la politique Autorisations définies dans cette politique, puis choisissez Créer une politique pour enregistrer votre travail.

Une fois que vous avez créé une politique, vous pouvez l'attacher à vos utilisateurs, groupes ou rôles. Pour de plus amples informations, veuillez consulter Ajout et suppression d'autorisations basées sur l'identité IAM.

Création de politiques IAM (AWS CLI)

Une politique est une entité qui, lorsqu'elle est attachée une identité ou à une ressource, définit les autorisations de cette dernière. Vous pouvez utiliser le AWS CLI pour créer des politiques gérées par le client dans IAM. Les politiques gérées par le client sont des politiques autonomes que vous gérez dans votre propre Compte AWS. Selon les bonnes pratiques, nous vous recommandons d'utiliser IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sûres et fonctionnelles. En procédant à la validation de vos politiques, vous pouvez corriger toute erreur ou recommandation avant d'attacher les politiques aux identités (utilisateurs, groupes et rôles) dans votre Compte AWS.

Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

Création de politiques IAM (AWS CLI)

Vous pouvez créer une politique gérée par le client IAM ou une politique en ligne à l'aide de l'interface AWS Command Line Interface (AWS CLI).

Pour créer une politique gérée par le client (AWS CLI)

Utilisez la commande suivante :

- create-policy

Pour créer une politique en ligne pour une identité IAM (groupe, utilisateur ou rôle) (AWS CLI)

Utilisez l'une des commandes suivantes :

- put-group-policy
- put-role-policy
- put-user-policy

> **Note**
>
> Vous ne pouvez pas utiliser IAM pour intégrer une politique en ligne pour un rôle lié à un service.

Pour valider une politique gérée par le client (AWS CLI)

Utilisez la commande IAM Access Analyzer suivante :

- validate-policy

Création de politiques IAM (AWS API)

Une politique est une entité qui, lorsqu'elle est attachée une identité ou à une ressource, définit les autorisations de cette dernière. Vous pouvez utiliser l' AWS API pour créer des politiques gérées par le client dans IAM. Les politiques gérées par le client sont des politiques autonomes que vous gérez dans votre propre Compte AWS. Selon les bonnes pratiques, nous vous recommandons d'utiliser IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sûres et fonctionnelles. En procédant à la validation de vos politiques, vous pouvez corriger toute erreur ou recommandation avant d'attacher les politiques aux identités (utilisateurs, groupes et rôles) dans votre Compte AWS.

Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

Création de politiques IAM (AWS API)

Vous pouvez créer une politique gérée par le client IAM ou une politique en ligne à l'aide de l'API AWS .

Pour créer une politique gérée par le client (AWS API)

Appelez l'opération suivante :

- CreatePolicy

Pour créer une politique intégrée pour une identité IAM (groupe, utilisateur ou rôle) (AWS API)

Appelez l'une des opérations suivantes :

- PutGroupPolicy
- PutRolePolicy
- PutUserPolicy

> **Note**
>
> Vous ne pouvez pas utiliser IAM pour intégrer une politique en ligne pour un rôle lié à un service.

Pour valider une politique gérée par le client (AWS API)

Appelez l'opération IAM Access Analyzer suivante :

- ValidatePolicy

Validation de politique IAM

Une politique est un document JSON écrit à l'aide de la syntaxe des politiques IAM. Lorsque vous attachez une politique à une entité IAM, telle qu'un utilisateur, un groupe ou un rôle, elle octroie des autorisations à cette entité.

Lorsque vous créez ou modifiez des politiques de contrôle d'accès IAM à l'aide du Console de gestion AWS, les examine AWS automatiquement pour s'assurer qu'elles sont conformes à la grammaire des politiques IAM. Si AWS détermine qu'une politique n'est pas conforme à la syntaxe, il vous invite à corriger la politique.

IAM Access Analyzer fournit des vérifications de politiques supplémentaires avec des recommandations pour vous aider à affiner davantage la politique. Pour en savoir plus sur les vérifications des politiques IAM Access Analyzer et les recommandations exploitables, veuillez

consulter Validation de politique IAM Access Analyzer. Pour afficher la liste des avertissements, erreurs et suggestions renvoyés par IAM Access Analyzer, veuillez consulter la Référence de vérification de politique IAM Access Analyzer.

Portée de la validation

AWS vérifie la syntaxe et la grammaire des politiques JSON. Il vérifie également que vous ARNs êtes correctement formaté et que les noms des actions et les clés de condition sont corrects.

Accès à la validation des politiques

Les politiques sont validées automatiquement lorsque vous créez une politique JSON ou que vous modifiez une politique existante dans la Console de gestion AWS. Si la syntaxe de la politique n'est pas valide, vous recevez une notification pour résoudre le problème avant de continuer. Les résultats de la validation de la politique d'IAM Access Analyzer sont automatiquement renvoyés dans le fichier Console de gestion AWS si vous disposez d'autorisations pour. `access-analyzer:ValidatePolicy` Vous pouvez également valider les politiques à l'aide de l' AWS API ou AWS CLI.

Politiques existantes

Il peut arriver que des politiques existantes ne soient pas valides, car leur création ou leur enregistrement sont antérieurs aux dernières mises à jour du moteur de politique. Selon les bonnes pratiques, nous vous recommandons d'utiliser IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sûres et fonctionnelles. Nous vous recommandons d'ouvrir vos politiques existantes et d'examiner les résultats de validation des politiques qui sont générés. Vous ne pouvez pas modifier et enregistrer des politiques existantes sans corriger les erreurs de syntaxe qu'elles contiennent.

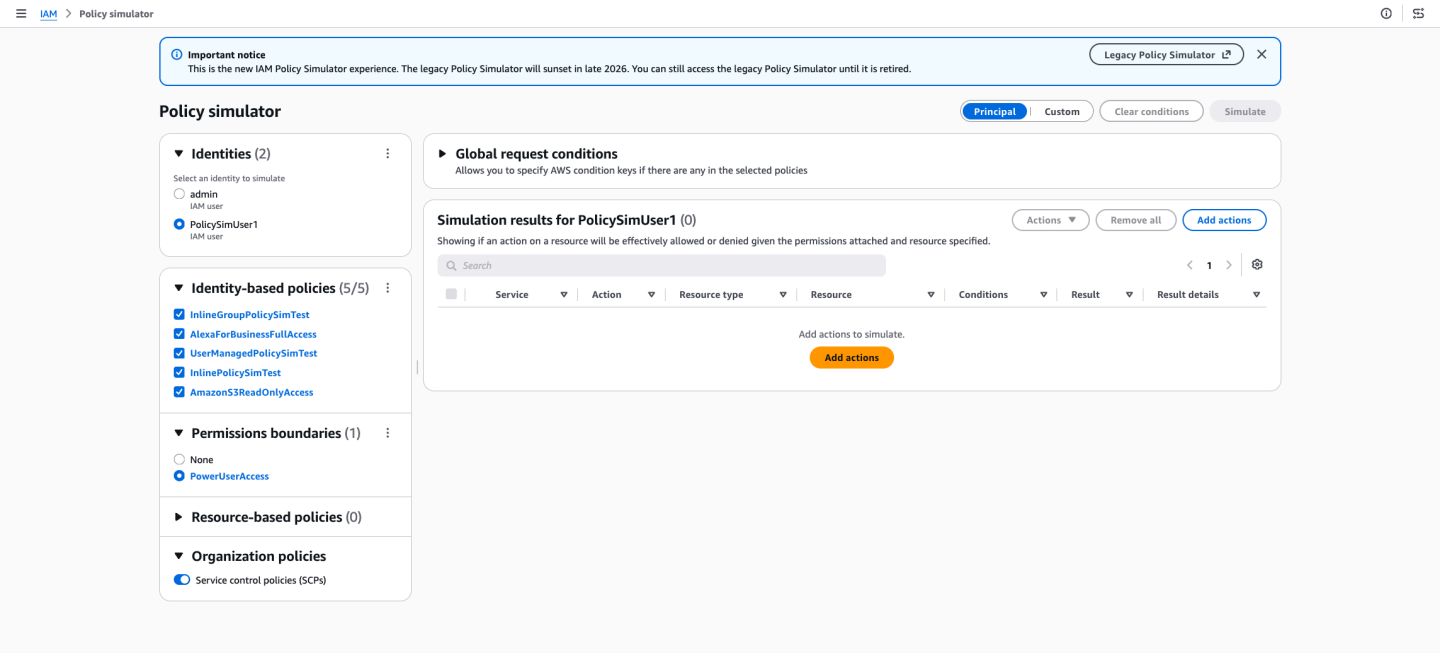
Test de politique IAM avec le simulateur de politiques IAM

Avec le simulateur de politiques IAM, vous pouvez tester les politiques basées sur l'identité, les limites d'autorisations IAM, les politiques de contrôle des services (SCP) et les politiques basées sur les ressources que vous fournissez. Vous simulez la manière dont une demande AWS serait évaluée par rapport à ces politiques, sans envoyer de véritable demande à aucun AWS service. Chaque simulation renvoie un résultat d'autorisation binaire pour chaque action et ressource que vous testez : l'action est autorisée ou refusée. Lorsque le résultat est une autorisation ou un refus explicite, le simulateur vous indique également quelle politique a produit ce résultat, afin que vous puissiez en comprendre la cause et y remédier.

Pour plus d'informations sur la manière d'utiliser les politiques IAM et pourquoi, consultez Politiques et autorisations dans Gestion des identités et des accès AWS.

> **Important**
>
> Les résultats du simulateur de politique peuvent différer de ceux de votre AWS environnement réel. Nous vous recommandons de comparer vos politiques à votre AWS environnement réel après les avoir testées avec le simulateur de politiques afin de confirmer que vous avez obtenu les résultats souhaités. Pour de plus amples informations, veuillez consulter Fonctionnement du simulateur de politiques IAM.

La capture d'écran suivante montre le simulateur de politiques IAM, dans lequel vous sélectionnez une identité et ses politiques, puis choisissez les actions et les ressources à tester avant d'exécuter la simulation.



Rubriques

- Modes de simulation de politiques
- Ce que vous pouvez faire avec le simulateur de politiques
- Fonctionnement du simulateur de politiques IAM
- Comment fonctionne l'évaluation des clés conditionnelles dans le simulateur de politique IAM
- Autorisations d'utilisation du simulateur de politiques

- Comment simuler des politiques

Modes de simulation de politiques

Le simulateur de politique comporte deux modes. Choisissez le mode qui vous convient si vous souhaitez tester les politiques qui existent déjà dans votre compte ou les politiques que vous êtes encore en train de rédiger.

Mode principal

En mode principal, vous testez les politiques déjà associées à un utilisateur, un rôle ou un groupe IAM existant. Vous choisissez une identité pour la simulation, et le simulateur évalue les politiques associées à cette identité. Vous pouvez également inclure ou exclure des politiques d'identité personnalisées ou des limites d'autorisations dans la simulation. Ces ajouts sont simulés comme s'ils étaient rattachés au principal, sans être réellement associés dans IAM, de sorte que votre compte reste inchangé.

Mode personnalisé

En mode Personnalisé, vous testez une ou plusieurs politiques que vous écrivez ou collez. Ceci est utile pour vérifier une politique avant de la joindre. Les politiques personnalisées sont simulées par elles-mêmes et ne sont pas enregistrées sur votre compte.

Pour obtenir des instructions détaillées, étape par étape, sur la sélection d'un mode et l'exécution d'une simulation, consultezComment simuler des politiques.

Ce que vous pouvez faire avec le simulateur de politiques

Voici quelques opérations courantes que vous pouvez effectuer à l'aide du simulateur de politiques. Vous trouverez un guide étape par étape plus détaillé et une distinction entre les comportements de l'API et de la console dansComment simuler des politiques.

- Testez les politiques basées sur l'identité, qu'elles soient déjà associées à un utilisateur, à un groupe ou à un rôle IAM, ou que vous écrivez ou collez de nouvelles politiques. Pour une identité associée à plusieurs politiques, vous pouvez tester toutes les politiques ensemble ou sélectionner des politiques individuelles, et voir quelles actions sont autorisées ou refusées pour des ressources spécifiques. Les nouvelles politiques sont évaluées comme si elles étaient associées à une identité, sans les associer réellement à votre compte ; elles ne sont utilisées que dans la simulation et ne sont pas enregistrées sur votre compte.

- Testez l'effet des contrôles restrictifs sur l'accès. Vous pouvez simuler une limite d'autorisations à la fois. Si vous Compte AWS êtes membre d'une organisation en AWS Organizations, vous pouvez également tester l'impact des politiques de contrôle des services (SCP) sur vos politiques basées sur l'identité. Le simulateur vous indique si un SCP autorise ou refuse une action, mais pour des raisons de sécurité, il n'affiche pas les instructions correspondantes d'un SCP comme il le fait pour les autres types de politiques.
- Testez l'effet d'une politique basée sur les ressources que vous fournissez. À l'exception de la console, le simulateur ne récupère pas la politique d'une ressource pour vous. Pour inclure une politique basée sur les ressources dans une simulation, vous devez fournir la politique en même temps que la ressource.
- Testez des services, des actions et des ressources spécifiques dans des conditions réelles et identifiez ce qui détermine le résultat. Par exemple, vous pouvez tester si une politique autorise les `DeleteBucket` actions `ListAllMyBucketsCreateBucket`, et dans Amazon S3 sur un compartiment spécifique, et fournir des clés de contexte telles qu'une adresse IP ou une date qui sont référencées dans les `Condition` éléments des politiques testées. Pour chaque résultat, le simulateur identifie l'instruction spécifique qui autorise ou refuse l'accès ; les instructions correspondantes peuvent provenir de politiques basées sur l'identité et de politiques basées sur les ressources que vous fournissez.

Fonctionnement du simulateur de politiques IAM

Lorsque vous exécutez une simulation, le simulateur de politique évalue les politiques visées par la demande, les éventuels SCP concernés et les entrées que vous fournissez, telles que les actions, les ressources et les valeurs clés contextuelles. En mode Personnalisé, vous fournissez directement les politiques. En mode principal, le simulateur utilise les politiques associées à l'identité que vous sélectionnez, et vous pouvez inclure ou exclure d'autres politiques dans la simulation. Le résultat est un résultat binaire pour chaque action et ressource : autorisé ou refusé.

Le simulateur de politique diffère de l' AWS environnement réel des manières suivantes :

- Le simulateur de politique n'émet pas de véritable demande de AWS service. Vous pouvez donc tester en toute sécurité les demandes qui pourraient autrement apporter des modifications indésirables à votre AWS environnement réel. Il n'utilise pas les valeurs clés de contexte réelles issues de la production et, comme les actions ne sont pas réellement exécutées, il ne renvoie aucune réponse de service. Le seul résultat est de savoir si chaque action demandée sera autorisée ou refusée.

- Si vous modifiez une politique dans le simulateur de politique, la modification n'affecte que la simulation. La police d'assurance correspondante dans votre contrat Compte AWS reste inchangée.
- Le simulateur de politique évalue les SCP, y compris les clés de condition et la portée des ressources dans les instructions de refus, mais ne prend pas en charge les politiques de contrôle des ressources (RCP). Les résultats de simulation peuvent toujours différer du comportement réel pour certaines configurations avancées, telles que les politiques de point de terminaison VPC, le chaînage de rôles ou plusieurs politiques basées sur des ressources sur une seule ressource.
- Cross-account les simulations renvoient les décisions par type de politique (politique d'identité et politique de ressources) dans. `EvalDecisionDetails`

Comment fonctionne l'évaluation des clés conditionnelles dans le simulateur de politique IAM

De nombreuses politiques utilisent `Condition` des éléments qui dépendent de clés de condition, tels que `aws:RequestedRegion` ou`aws:MultiFactorAuthPresent`. Lorsque vous exécutez une simulation, le simulateur de politique obtient les valeurs des clés de condition à partir de deux sources : les valeurs que le simulateur remplit automatiquement et les valeurs que vous fournissez en tant qu'entrée de simulation. Si une valeur ne satisfait pas à une condition, l'action est refusée.

Contexte que le simulateur remplit automatiquement

Le simulateur renseigne les clés contextuelles principales et organisationnelles suivantes pour l'évaluation :

- Contexte principal : `aws:PrincipalAccount`aws:PrincipalId,`aws:PrincipalType`,`aws:Type`,`aws:UserId`, `aws:UserName`
- Contexte de l'organisation :`aws:PrincipalOrgID`,`aws:PrincipalOrgMasterAccountId`, `aws:PrincipalOrgPaths`

Comme ces clés sont renseignées pour vous, les SCP et autres politiques qui utilisent des conditions de principal ou d'organisation sont correctement évalués. Vous fournissez les valeurs de toutes les autres clés de condition auxquelles vos politiques font référence. Passez en revue les `Condition` éléments des politiques que vous testez et fournissez des valeurs pour toutes les clés qui ne figurent pas dans la liste ci-dessus.

Clés de condition globales et assistance technique

Pour les politiques basées sur l'identité et les politiques basées sur les ressources, le simulateur ne détermine pas quels services prennent en charge une clé de condition globale donnée pour l'autorisation. Par exemple, le simulateur n'indique pas qu'un service ne prend pas en charge`aws:TagKeys`. En revanche, les politiques de contrôle des services (SCP) sont évaluées à l'aide d'une logique tenant compte des services, y compris des clés de condition globales telles que et. `aws:RequestedRegion` `aws:PrincipalAccount` Pour plus d'informations sur les clés de condition, consultez AWS clés de contexte de condition globale.

Clés de condition dans les politiques de contrôle des services (SCP)

Le simulateur évalue les clés de condition et la portée des ressources dans les SCP. La manière dont vous fournissez les valeurs pour ces clés dépend de la manière dont vous exécutez la simulation :

- Avec l' AWS API AWS CLI or, vous pouvez fournir des valeurs pour les clés de condition référencées par les SCP.
- Dans la console IAM, vous pouvez uniquement définir des valeurs pour les clés de condition qui apparaissent dans vos politiques basées sur l'identité, les limites d'autorisations ou les politiques basées sur les ressources. Si une clé de condition est référencée uniquement par un SCP, vous ne pouvez pas définir de valeur pour celle-ci dans la console. Toutefois, si la même clé de condition apparaît également dans l'une de ces autres politiques, la valeur que vous définissez est utilisée pour l'ensemble de l'évaluation, y compris le SCP.

Pour des raisons de sécurité, l'évaluation SCP ne renvoie pas les valeurs de contexte manquantes. Vous recevez toujours la décision d'autoriser ou de refuser la demande.

Autorisations d'utilisation du simulateur de politiques

Les autorisations dont vous avez besoin varient selon que vous utilisez l' AWS API AWS CLI or ou la console IAM, et selon le mode que vous utilisez.

Console

Vous devez être autorisé à appeler `iam:GetContextKeysForCustomPolicy` pour utiliser le simulateur de politique IAM sur la console.

Les autorisations supplémentaires dont vous avez besoin dépendent du mode. En mode Principal, vous avez besoin d'autorisations de lecture supplémentaires pour répertorier les

identités de votre compte et lire les politiques qui y sont associées. Pour réduire le montant que vous devez accorder, utilisez le mode Personnalisé si vous devez uniquement tester les politiques que vous collez.

Mode personnalisé

Pour utiliser le mode personnalisé, dans lequel vous collez les politiques que vous souhaitez tester, vous devez également `iam:SimulateCustomPolicy`

Mode principal

Pour utiliser le mode Principal, dans lequel la console charge et affiche les identités et les politiques de votre compte, vous devez disposer de l'autorisation commune et des informations suivantes. Il s'agit de l'ensemble le plus important, car la console doit découvrir les identités et lire les politiques qui y sont associées.

Simuler :

- `iam:SimulatePrincipalPolicy`

Identités de la liste :

- `iam:ListUsers`, `iam:ListRoles`, `iam:ListGroups`
- `iam:ListGroupsForUser`

Liste des politiques ci-jointes :

- `iam:ListPolicies`, `iam:ListUserPolicies`, `iam:ListRolePolicies`, `iam:ListGroupPolicies`
- `iam:ListAttachedUserPolicies`, `iam:ListAttachedRolePolicies`, `iam:ListAttachedGroupPolicies`

Obtenir des documents de politique :

- `iam:GetPolicyVersion`, `iam:GetUserPolicy`, `iam:GetRolePolicy`, `iam:GetGroupPolicy`

Obtention des limites d'autorisations :

- `iam:GetUser`, `iam:GetRole`

Autorisations supplémentaires pour le mode Principal afin de simuler des politiques basées sur les ressources

Pour inclure la politique d'une ressource dans une simulation en mode Principal, ajoutez l'autorisation de lecture pour ce type de ressource :

- `s3:GetBucketPolicy`, `sns:GetTopicAttributes`, `sqs:GetQueueUrl`, `sqs:GetQueueAttributes`, `glacier:GetVaultAccessPolicy`

Pour des exemples de politiques qui autorisent l'accès au simulateur, consultezthe section called "Exemples de politiques : Gestion des identités et des accès AWS (IAM)".

AWS CLI and AWS API

Pour l' AWS API AWS CLI et, vous n'avez besoin que de l'autorisation d'appeler l'action de simulation pour le type de politique que vous souhaitez tester. Si vous utilisez `iam:GetContextKeysForCustomPolicy` et `iam:GetContextKeysForPrincipalPolicy` pour découvrir les clés contextuelles en premier, accordez également des autorisations pour ces actions. Pour des raisons de sécurité, les actions sont réparties selon les deux modes suivants. Accordez uniquement les autorisations correspondant à ce que vous souhaitez simuler.

Pour simuler des politiques que vous transmettez directement à l'API sous forme de chaînes (politiques personnalisées)

- `iam:SimulateCustomPolicy`
- `iam:GetContextKeysForCustomPolicy`(uniquement si vous utilisez l'étape de découverte par clé contextuelle)

Ces actions vous permettent de tester des politiques qui ne sont pas encore associées à un utilisateur, un groupe d'utilisateurs ou un rôle. Vous transmettez les politiques sous forme de chaînes, et elles ne sont utilisées que dans la simulation.

Pour simuler des politiques associées à un utilisateur, un groupe d'utilisateurs, un rôle ou une ressource IAM

- `iam:SimulatePrincipalPolicy`

- `iam:GetContextKeysForPrincipalPolicy`(uniquement si vous utilisez l'étape de découverte par clé contextuelle)

Étant donné que ces actions peuvent révéler les autorisations attribuées à d'autres entités IAM, pensez à restreindre les personnes autorisées à les appeler. Par exemple, pour autoriser un utilisateur nommé Bob à simuler une politique attribuée à un utilisateur nommé Alice, autorisez Bob à accéder à la ressource`arn:aws:iam::777788889999:user/alice`.

Pour consulter un exemple de politique qui permet d'utiliser l'API du simulateur de politiques pour les politiques attachées et non attachées actuellement Compte AWS, consultezIAM : Accès à l'API du simulateur de politique. Pour afficher un exemple de politique applicable aux utilisateurs ayant un chemin spécifique, consultezIAM : accès à l'API du simulateur de politique en fonction du chemin d'utilisateur.

Comment simuler des politiques

Vous pouvez exécuter des simulations dans la console IAM ou à l'aide de l' AWS API AWS CLI et. Pour un aperçu de ce que le simulateur évalue, des deux modes et de la manière dont il obtient un résultat, voirTest de politique IAM avec le simulateur de politiques IAM. Pour connaître les autorisations dont vous avez besoin, consultezAutorisations d'utilisation du simulateur de politiques.

Rubriques

- Simuler des politiques (console)
- Simuler des politiques (AWS CLI and AWS API)

Simuler des politiques (console)

Vous pouvez exécuter des simulations à partir du simulateur de politiques de la console https://console.aws.amazon.com/iam/ IAM. Dans le volet de navigation, choisissez Policy simulator.

> **Note**
>
> La console autonome du simulateur de politiques n'https://policysim.aws.amazon.com/est plus maintenue. Utilisez le simulateur de politique dans la console IAM.

Mode principal

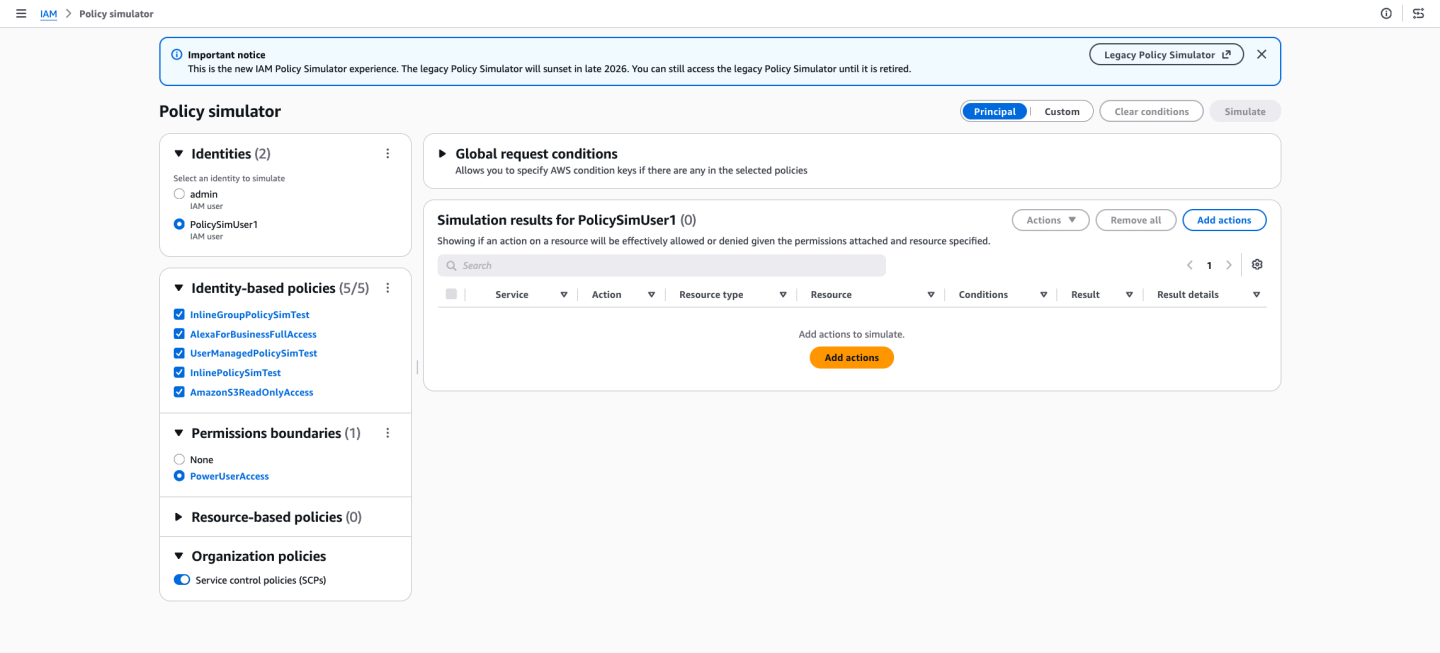
Sélectionnez une identité

Vous commencez par sélectionner une identité. Vous pouvez sélectionner des utilisateurs, des rôles ou des groupes IAM dans ce compte. Le panneau latéral renseignera les politiques associées à ces identités après votre sélection.

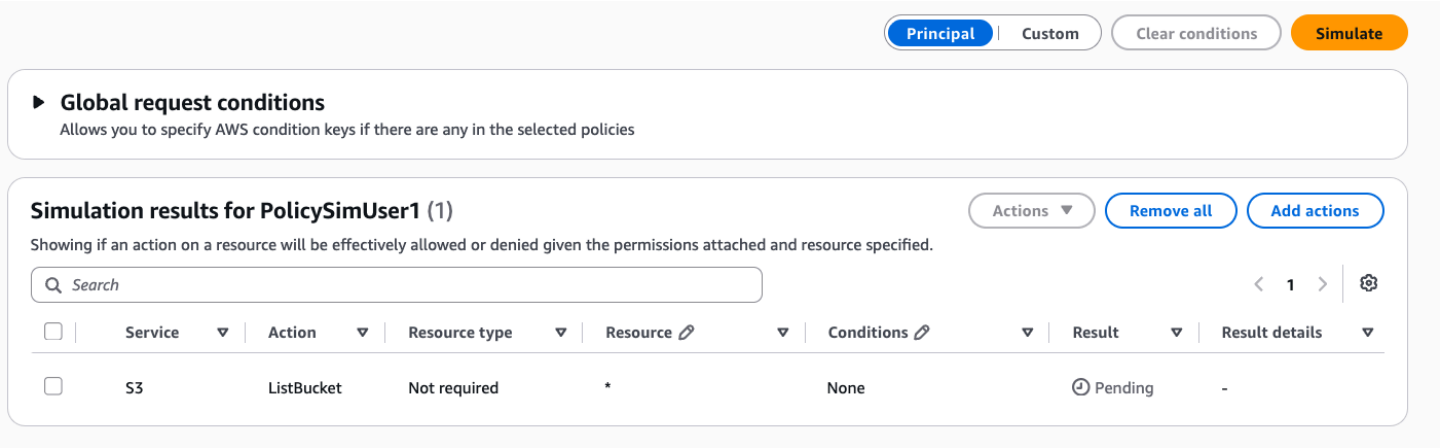
Inclure ou exclure des politiques

Le panneau latéral répertorie les politiques concernées par la simulation, y compris les politiques en ligne, AWS gérées et gérées par le client, ainsi que toute limite d'autorisations. Vous pouvez exclure une politique pour voir comment le résultat change sans elle, puis l'inclure à nouveau. L'exclusion d'une politique qui a accordé l'accès change le résultat en un refus implicite.

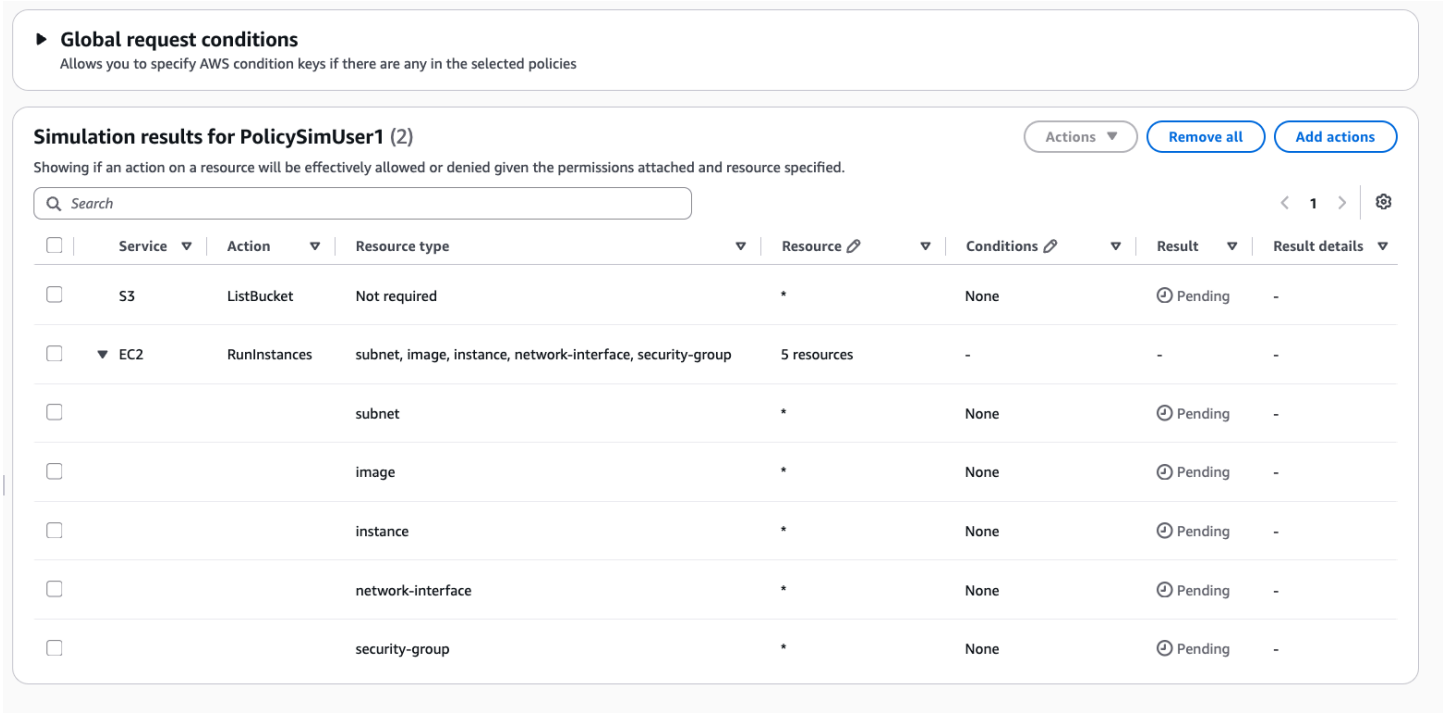
Ajouter des actions et des ressources



Après avoir choisi les identités et les politiques dans le panneau de gauche, ajoutez les actions que vous souhaitez tester. Pour une action qui prend en charge une sélection de ressources, vous pouvez effectuer un test par rapport à toutes les ressources ou saisir un ou plusieurs ARN de ressources spécifiques. Lorsque plusieurs actions utilisent le même type de ressource, vous pouvez appliquer une ressource à toutes à la fois ou la limiter à une seule action.



Certaines actions nécessitent un groupe de types de ressources connexes avant que le simulateur puisse les évaluer. Par exemple, lorsque vous effectuez une simulation`ec2:RunInstances`, la console vous invite à spécifier ensemble une image, une instance et un groupe de sécurité, ainsi qu'un sous-réseau ou un volume selon le scénario.



Fournir des valeurs clés de condition

Si les politiques que vous testez incluent des clés de condition, le simulateur répertorie les clés pertinentes et vous permet de définir des valeurs avant d'exécuter la simulation. Les clés affichées reflètent les politiques actuellement sélectionnées. Le fait de sélectionner ou de désactiver une politique ajoute ou supprime ses clés. Si une valeur ne satisfait pas à une condition, l'action est refusée.

Le simulateur fournit des valeurs pour les clés de condition à deux endroits. Les conditions de demande globales s'appliquent à l'ensemble de la demande simulée. Il s'agit des touches commençant par`aws:`, par exemple, `aws:MultiFactorAuthPresent` ou`aws:PrincipalServiceNamesList`, et vous définissez leurs valeurs une fois dans la zone des conditions globales. La sélection ou la désactivation d'une politique ajoute ou supprime ses clés globales, et les valeurs que vous saisissez sont conservées lorsque vous ajustez les politiques concernées. Service-specific certaines conditions s'appliquent à une seule action. Chaque ligne d'action indique le nombre de conditions auxquelles elle fait référence, et vous ouvrez la ligne pour définir les clés spécifiques au service pour cette action, par exemple `cloudwatch:namespace` pour `cloudwatch:PutMetricData` ou `ram:PermissionResourceType` pour. `ram:ListPermissionAssociations` Une ligne d'action peut inclure à la fois une clé globale et une clé spécifique à un service, et si une valeur ne satisfait pas à une condition, cette action est refusée.

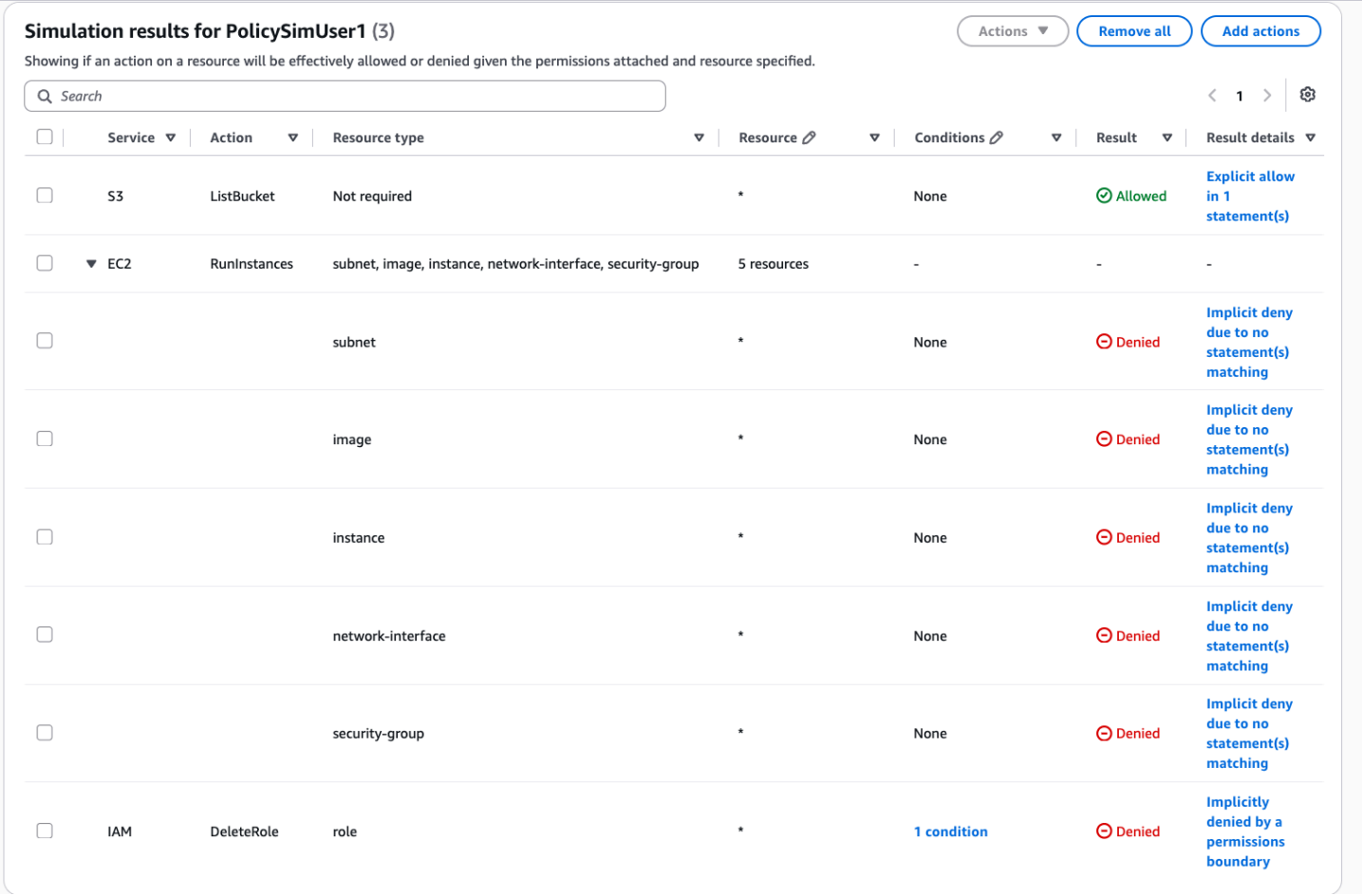
Le simulateur de politique IAM remplit certaines clés contextuelles principales et organisationnelles pour vous, et vous fournissez des valeurs pour le reste. Pour obtenir la liste des clés qui sont renseignées automatiquement et pour en savoir plus sur la façon dont les clés de condition sont évaluées, consultezComment fonctionne l'évaluation des clés conditionnelles dans le simulateur de politique IAM.

Pour savoir comment utiliser les touches de condition, consultezAWS clés de contexte de condition globale.

Tester une limite d'autorisations

Lorsqu'une identité possède une limite d'autorisations, celle-ci apparaît dans le panneau latéral et est incluse par défaut. Vous pouvez également exclure la limite et relancer la simulation si vous souhaitez savoir si la limite est la raison pour laquelle une action est refusée.

Supposons, par exemple, que vous `iam:DeleteRole` simuliez une identité dont les limites ne le permettent pas. Le résultat est un refus attribué à la limite des autorisations. Si vous ajoutez ensuite une politique qui autorise l'action et exclut la limite, le résultat devient une autorisation. Cela confirme que la limite était le seul contrôle bloquant l'action.



Incluez des politiques de contrôle des services

Si votre compte appartient à une organisation, vous pouvez inclure des politiques de contrôle des services (SCP) dans la simulation depuis le panneau latéral. Le simulateur évalue les clés de condition remplies automatiquement et la portée des ressources dans les SCP lorsqu'il détermine le résultat. Lorsqu'un SCP refuse explicitement une action, il en résulte un refus explicite attribué à AWS Organizations, et une autorisation dans une autre politique ne peut pas la remplacer. Désactivez le contrôle et relancez la simulation pour voir le résultat sans les contrôles de l'organisation. Pour obtenir la liste des clés de condition remplies automatiquement, consultezComment fonctionne l'évaluation des clés conditionnelles dans le simulateur de politique IAM.

> **Note**
>
> Pour des raisons de sécurité, le simulateur n'affiche pas les détails de l'évaluation du SCP. Les clés de condition référencées uniquement par les SCP ne sont pas répertoriées pour que vous puissiez définir des valeurs, et les instructions correspondantes renvoyées pour

un résultat n'incluent pas les instructions provenant des SCP. Les clés et les déclarations correspondantes proviennent des politiques basées sur l'identité associées à l'identité et de toutes les politiques que vous fournissez.

Testez une politique basée sur les ressources

Pour inclure une politique basée sur les ressources dans une simulation de console, entrez un ARN de ressource spécifique et choisissez l'option permettant d'inclure la politique de la ressource. La politique basée sur les ressources apparaît dans le panneau latéral et est évaluée dans le résultat. Dans la console, cela est disponible pour les compartiments Amazon S3, les files d'attente Amazon SQS, les rubriques Amazon SNS et les coffres-forts Amazon Glacier.

> **Note**
>
> La simulation de politiques basées sur les ressources n'est pas prise en charge pour les rôles IAM.

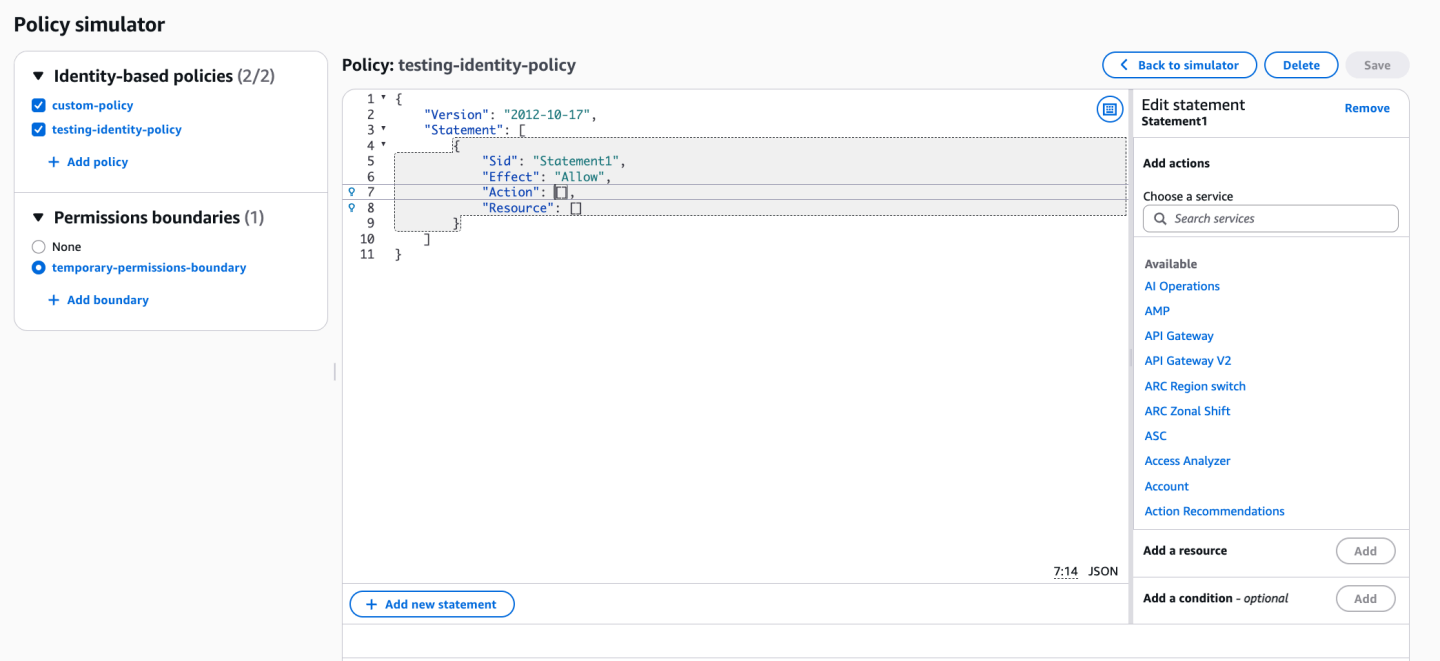
Lire les résultats

Chaque ligne du tableau des résultats indique si l'action est autorisée, implicitement refusée ou explicitement refusée, ainsi qu'une brève explication du résultat. Utilisez l'explication pour identifier la politique ou le contrôle qui a déterminé le résultat, par exemple une autorisation explicite, l'absence d'instruction correspondante, une limite d'autorisations ou un contrôle d'organisation.

Mode personnalisé

Vous pouvez utiliser l'éditeur de politiques IAM pour écrire ou coller des politiques basées sur l'identité et des limites d'autorisation en mode personnalisé. Ces politiques ne sont pas associées à une identité et peuvent être évaluées avant de les appliquer. Vous pouvez inclure plusieurs politiques basées sur l'identité, mais une seule limite d'autorisations pour la simulation.

> **Note**
>
> En mode personnalisé, la console ne prend pas en charge la transmission de SCP personnalisés ni la simulation de politiques basées sur les ressources. Pour simuler des SCP personnalisés ou des politiques basées sur les ressources, utilisez l' AWS CLI API or. AWS



Simuler des politiques (AWS CLI and AWS API)

La simulation de politiques à l'aide de l' AWS API AWS CLI or se fait généralement en deux étapes :

1. Évaluez les politiques et renvoyez la liste des clés de contexte auxquelles elles font référence, afin de savoir quelles valeurs de clé de contexte fournir. Cette étape est facultative.
2. Simulez les politiques en fournissant les actions, les ressources et les valeurs clés contextuelles à utiliser lors de la simulation.

Chaque action est associée à chaque ressource, et la simulation détermine si les politiques autorisent ou refusent cette action pour cette ressource. Vous pouvez fournir des valeurs pour toutes les clés de contexte auxquelles vos politiques font référence. Si vous ne fournissez pas de valeur pour une clé de contexte, la simulation s'exécute toujours et renvoie un refus implicite, et les clés manquantes sont renvoyées `missingContextValues` afin que vous sachiez quelles valeurs ajouter. Si une clé de contexte manquante est référencée uniquement par un SCP, cette clé n'est pas renvoyée pour des raisons de sécurité, mais vous recevez tout de même la décision.

Les actions de l'API sont divisées en deux groupes :

- Actions qui simulent uniquement les politiques transmises directement à l'API sous forme de chaînes : GetContextKeysForCustomPolicy et SimulateCustomPolicy.
- Actions qui simulent les politiques associées à un utilisateur, un groupe d'utilisateurs, un rôle ou une ressource IAM spécifié : GetContextKeysForPrincipalPolicy et SimulatePrincipalPolicy. Étant

donné que ceux-ci peuvent révéler les autorisations attribuées à d'autres entités IAM, pensez à restreindre l'accès à celles-ci.

Ces actions soutiennent également les objectifs suivants :

- `OrderedOrganizationPolicyInputListo`n vous SimulateCustomPolicy permet de simuler l'effet de la hiérarchie SCP d'une organisation sans utiliser de principal actif.
- `PolicyExclusionListo`n vous SimulatePrincipalPolicy permet de tester « Et si je supprimais cette politique ? » scénarios sans modifier les politiques associées au principal.
- La réponse en renvoie une `EvaluationResult` par action, avec les décisions agrégées au niveau supérieur et les détails par ressource dans`ResourceSpecificResults`.

AWS CLI and AWS Référence des API

Pour des exemples de AWS CLI commandes permettant de simuler des politiques, consultez les rubriques suivantes :

- AWS CLI: obtenir des clés de contexte pour une politique personnalisée, obtenir des clés de contexte pour une politique principale, simuler une politique personnalisée, simuler une politique principale https://docs.aws.amazon.com/cli/latest/reference/iam/get-context-keys-for-principal-policy.html https://docs.aws.amazon.com/cli/latest/reference/iam/simulate-custom-policy.html https://docs.aws.amazon.com/cli/latest/reference/iam/simulate-principal-policy.html
- AWS API : GetContextKeysForCustomPolicy, GetContextKeysForPrincipalPolicy, SimulateCustomPolicy, SimulatePrincipalPolicy

Ajout et suppression d'autorisations basées sur l'identité IAM

Vous utilisez des politiques pour définir les autorisations pour une identité (utilisateur, groupe d'utilisateurs ou rôle). Vous pouvez ajouter et supprimer des autorisations en attachant et en détachant des politiques IAM pour une identité à l'aide de l'API Console de gestion AWS, du AWS Command Line Interface (AWS CLI) ou de l' AWS API. Vous pouvez aussi utiliser des politiques afin de définir les limites d'autorisations uniquement pour les entités (utilisateurs ou rôles) qui utilisent les mêmes méthodes. Les limites d'autorisations sont une AWS fonctionnalité avancée qui contrôle le maximum d'autorisations qu'une entité peut avoir.

Rubriques

- Terminologie
- Afficher l'activité d'identité
- Ajout des autorisations d'identité IAM (console)
- Suppression des autorisations d'identité IAM (console)
- Ajout de politiques IAM (AWS CLI)
- Supprimer des politiques IAM (AWS CLI)
- Ajouter des politiques IAM (AWS API)
- Supprimer les politiques IAM (AWS API)

Terminologie

Lorsque vous associez des politiques d'autorisations à des identités (utilisateurs IAM, groupes IAM et rôles IAM), la terminologie et les procédures varient selon que vous utilisez ou pas une politique gérée ou en ligne :

- Attach (Attacher) : utilisé avec les politiques gérées. Vous attachez une politique gérée à une identité (utilisateur, groupe d'utilisateurs ou rôle). L'attachement d'une politique applique les autorisations de cette dernière à l'identité.
- Detach (Détacher) : utilisé avec les politiques gérées. Vous détachez une politique gérée d'une identité IAM (utilisateur, groupe d'utilisateurs ou rôle). Le détachement d'une politique entraîne la suppression des ses autorisations de l'identité.
- Embed (Intégrer) : utilisé avec les politiques en ligne. Vous intégrez une politique en ligne à une identité (utilisateur, groupe d'utilisateurs ou rôle). L'intégration d'une politique applique les autorisations de cette dernière à l'identité. Dans la mesure où une politique en ligne est stockée dans l'identité, elle est intégrée plutôt qu'attachée, même si les résultats sont similaires.

  > **Note**
  >
  > Vous ne pouvez intégrer une politique en ligne pour un rôle lié à un service que dans le service qui dépend du rôle. Veuillez consulter la documentation AWS de votre service pour savoir si celui-ci prend en charge cette fonction.

- Delete (Supprimer) : utilisé avec les politiques en ligne. Vous supprimez une politique en ligne d'une identité IAM (utilisateur, groupe d'utilisateurs ou rôle). La suppression d'une politique entraîne la suppression de ses autorisations de l'identité.

> **Note**
>
> Vous ne pouvez supprimer une politique en ligne pour un rôle lié à un service que dans le service qui dépend du rôle. Veuillez consulter la documentation AWS de votre service pour savoir si celui-ci prend en charge cette fonction.

Vous pouvez utiliser la console ou AWS CLI l' AWS API pour effectuer l'une de ces actions.

En savoir plus

- Pour en savoir plus sur la différence entre les stratégies gérées et les stratégies en ligne, consultez Politiques gérées et politiques en ligne.
- Pour plus d'informations sur les limites d'autorisations, consultez Limites d'autorisations pour les entités IAM.
- Pour obtenir des informations d'ordre général sur les politiques IAM, consultez Politiques et autorisations dans Gestion des identités et des accès AWS.
- Pour plus d'informations sur la validation des politiques IAM, veuillez consulter Validation de politique IAM.
- Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

Afficher l'activité d'identité

Avant de modifier les autorisations d'une identité (utilisateur, groupe d'utilisateurs ou rôle), vous devez examiner leur activité récente au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Ajout des autorisations d'identité IAM (console)

Vous pouvez utiliser le Console de gestion AWS pour ajouter des autorisations à une identité (utilisateur, groupe d'utilisateurs ou rôle). Pour ce faire, attachez les stratégies gérées qui contrôlent les autorisations, ou spécifiez une stratégie qui sert de limite d'autorisations. Vous pouvez également intégrer une politique en ligne.

Pour utiliser une politique gérée en tant que politique d'autorisations pour une identité (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des politiques, sélectionnez la case d'option en regard du nom de la politique à attacher. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Sélectionnez Actions, puis Attach (Attacher).
5. Sélectionnez une ou plusieurs identités auxquelles attacher la politique. Vous pouvez utiliser la zone de recherche pour filtrer la liste des entités principales. Après avoir sélectionné les identités, choisissez Attacher une politique.

Pour utiliser une politique gérée permettant de définir une limite d'autorisations (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de celle à modifier. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Sur la page des détails de la politique, choisissez l'onglet Entités attachées, puis, si nécessaire, ouvrez la section Attachées en tant que limites des autorisations et choisissez Définir cette politique en tant que limite des autorisations.
5. Sélectionnez un ou plusieurs utilisateurs ou rôles sur lesquels utiliser la politique comme limite d'autorisations. Vous pouvez utiliser la zone de recherche pour filtrer la liste des entités principales. Après avoir sélectionné les principaux, choisissez Définir la limite des autorisations.

Pour intégrer une politique en ligne pour un utilisateur ou un rôle (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, sélectionnez Users (Utilisateurs) ou Roles (Rôles).
3. Dans la liste, sélectionnez le nom de l'utilisateur ou du rôle auquel intégrer une politique.
4. Sélectionnez l’onglet Autorisations.

5. Sélectionnez Ajouter des autorisations, puis Ajouter la politique.

> **Note**
>
> Vous ne pouvez pas intégrer une politique en ligne dans un rôle lié à un service dans IAM. Comme le service lié définit si vous pouvez modifier les autorisations du rôle, vous pourrez peut-être ajouter des politiques depuis la console de service, l'API ou l'interface AWS CLI. Pour afficher la documentation d'un rôle lié à un service, consultez AWS services qui fonctionnent avec IAM et choisissez Oui dans la colonne Rôle lié à un service de votre service.

6. Faites votre choix parmi les méthodes suivantes pour afficher les étapes requises pour créer votre politique :

   - Importation de politiques gérées existantes : vous pouvez importer une politique gérée dans votre compte, puis la modifier afin de la personnaliser en fonction des vos exigences spécifiques. Une politique gérée peut être une politique AWS gérée ou une politique gérée par le client que vous avez créée précédemment.
   - Création de politiques avec l'éditeur visuel : vous pouvez construire intégralement une nouvelle politique dans l'éditeur visuel. Si vous utilisez l'éditeur visuel, vous n'avez pas besoin de comprendre la syntaxe JSON.
   - Création de politiques à l'aide de l'éditeur JSON : dans l'option éditeur JSON, vous pouvez créer une politique à l'aide de la syntaxe JSON. Vous pouvez composer un nouveau document de stratégie JSON ou coller un exemple de stratégie.

7. Une fois que vous avez créé une politique en, ligne, elle est automatiquement intégrée à votre utilisateur ou rôle.

Pour intégrer une politique en ligne pour un groupe d'utilisateurs (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs).
3. Dans la liste, sélectionnez le nom du groupe d'utilisateurs auquel intégrer une politique.
4. Choisissez l'onglet Permissions (Autorisations), puis Add permissions (Ajouter des autorisations), et enfin Create inline policy (Créer une politique en ligne).

5. Effectuez l'une des actions suivantes :
   - Sélectionnez l'option Visuel pour créer la politique. Pour de plus amples informations, veuillez consulter Création de politiques avec l'éditeur visuel.
   - Sélectionnez l'option JSON pour créer la politique. Pour de plus amples informations, veuillez consulter Création de politiques à l'aide de l'éditeur JSON.
6. Lorsque vous êtes satisfait de la politique, choisissez Créer une politique.

Pour modifier la limite d'autorisations d'une ou plusieurs entités (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de celle à modifier. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Sur la page des détails de la politique, choisissez l'onglet Entités attachées, puis, si nécessaire, ouvrez la section Attachées en tant que limite des autorisations. Cochez la case en regard des utilisateurs ou des rôles dont vous voulez modifier les limites, puis choisissez Modifier.
5. Sélectionnez une nouvelle politique à utiliser comme limite d'autorisations. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques. Après avoir sélectionné la politique, choisissez Définir la limite des autorisations.

Suppression des autorisations d'identité IAM (console)

Vous pouvez utiliser le Console de gestion AWS pour supprimer les autorisations associées à une identité (utilisateur, groupe d'utilisateurs ou rôle). Pour ce faire, détachez les politiques gérées qui contrôlent les autorisations, ou supprimez une politique qui a servi de limite d'autorisations. Vous pouvez également supprimer une politique en ligne.

Pour détacher une politique gérée utilisée en tant que politique d'autorisations (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.

3. Dans la liste des politiques, sélectionnez la case d'option en regard du nom de la politique à détacher. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Sélectionnez Actions, puis Detach (Détacher).
5. Sélectionnez les identités desquelles détacher la politique. Vous pouvez utiliser la zone de recherche pour filtrer la liste des identités. Après avoir sélectionné les identités, choisissez Détacher la politique.

Pour supprimer une limite d'autorisations (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de celle à modifier. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Sur la page de résumé de la politique, choisissez l'onglet Entités attachées, puis, si nécessaire, ouvrez la section Attachée en tant que limite des autorisations et choisissez les entités pour lesquelles vous supprimez la limite des autorisations. Puis, choisissez Supprimer la limite.
5. Confirmez la suppression de la limite et choisissez Supprimer la limite.

Pour supprimer une politique en ligne (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), Users (Utilisateurs) ou Roles (Rôles).
3. Choisissez dans la liste le nom du groupe d'utilisateurs, de l'utilisateur ou du rôle pour lequel vous souhaitez supprimer la politique.
4. Sélectionnez l'onglet Autorisations.
5. Cochez la case en regard de la politique et choisissez Supprimer.
6. Dans la boîte de confirmation, choisissez Supprimer.

Ajout de politiques IAM (AWS CLI)

Vous pouvez utiliser le AWS CLI pour ajouter des autorisations à une identité (utilisateur, groupe d'utilisateurs ou rôle). Pour ce faire, attachez les stratégies gérées qui contrôlent les autorisations, ou spécifiez une stratégie qui sert de limite d'autorisations. Vous pouvez également intégrer une politique en ligne.

Pour utiliser une politique gérée en tant que politique d'autorisations pour une entité (AWS CLI)

1. (Facultatif) Pour afficher les informations d'une politique gérée, exécutez les commandes suivantes :
   - Pour répertorier les stratégies gérées : aws iam list-policies
   - Pour récupérer les informations détaillées sur une politique gérée : get-policy
2. Pour attacher une politique gérée à une identité (utilisateur, groupe d'utilisateurs ou rôle), utilisez l'une des commandes suivantes :
   - était un objectif attach-user-policy
   - était un objectif attach-group-policy
   - était un objectif attach-role-policy

Pour utiliser une politique gérée permettant de définir une limite d'autorisations (AWS CLI)

1. (Facultatif) Pour afficher les informations d'une politique gérée, exécutez les commandes suivantes :
   - Pour répertorier les stratégies gérées : aws iam list-policies
   - Pour récupérer les informations détaillées sur une politique gérée : aws iam get-policy
2. Pour utiliser une politique gérée permettant de définir la limite d'autorisations pour une entité (utilisateur ou rôle), utilisez l'une des commandes suivantes :
   - était un objectif put-user-permissions-boundary
   - était un objectif put-role-permissions-boundary

Pour intégrer une politique en ligne (AWS CLI)

Pour intégrer une politique en ligne à une identité (utilisateur, groupe d'utilisateurs ou rôle qui n'est pas un rôle lié à un service), utilisez l'une des commandes suivantes :

- était un objectif put-user-policy
- était un objectif put-group-policy
- était un objectif put-role-policy

Supprimer des politiques IAM (AWS CLI)

Vous pouvez utiliser le AWS CLI pour détacher les politiques gérées qui contrôlent les autorisations ou supprimer une politique qui sert de limite d'autorisations. Vous pouvez également supprimer une politique en ligne.

Pour détacher une politique gérée utilisée en tant que politique d'autorisations (AWS CLI)

1. (Facultatif) Pour afficher des informations sur une politique, exécutez les commandes suivantes :
   - Pour répertorier les stratégies gérées : aws iam list-policies
   - Pour récupérer les informations détaillées sur une politique gérée : aws iam get-policy
2. (Facultatif) Pour en savoir plus sur les relations entre les stratégies et les identités, exécutez les commandes suivantes :
   - Pour répertorier les identités (utilisateurs IAM, groupes IAM et rôles IAM) auxquelles une politique gérée est attachée :
     - était un objectif list-entities-for-policy
   - Pour répertorier les politiques gérées attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle), utilisez l'une des commandes suivantes :
     - était un objectif list-attached-user-policies
     - était un objectif list-attached-group-policies
     - était un objectif list-attached-role-policies
3. Pour détacher une politique gérée d'une identité (utilisateur, groupe d'utilisateurs ou rôle), utilisez l'une des commandes suivantes :
   - était un objectif detach-user-policy
   - était un objectif detach-group-policy
   - était un objectif detach-role-policy

Pour supprimer une limite d'autorisations (AWS CLI)

1. (Facultatif)Pour afficher la politique gérée actuellement utilisée pour définir la limite d'autorisations d'un utilisateur ou rôle, exécutez les commandes suivantes :
   - aws iam get-user
   - aws iam get-role
2. (Facultatif)Pour afficher les utilisateurs ou rôles sur lesquels une politique gérée est utilisée pour une limite d'autorisations, exécutez la commande suivante :
   - était un objectif list-entities-for-policy
3. (Facultatif) Pour afficher les informations d'une politique gérée, exécutez les commandes suivantes :
   - Pour répertorier les stratégies gérées : aws iam list-policies
   - Pour récupérer les informations détaillées sur une politique gérée : aws iam get-policy
4. Pour supprimer une limite d'autorisations d'un utilisateur ou d'un rôle, utilisez l'une des commandes suivantes :
   - était un objectif delete-user-permissions-boundary
   - était un objectif delete-role-permissions-boundary

Pour supprimer une politique en ligne (AWS CLI)

1. (Facultatif) Pour répertorier toutes les politiques en ligne attachées à une identité (utilisateur, groupe d'utilisateurs, rôle), utilisez l'une des commandes suivantes :
   - était un objectif list-user-policies
   - était un objectif list-group-policies
   - était un objectif list-role-policies
2. (Facultatif) Pour récupérer un document de politique en ligne qui est intégré à une identité (utilisateur, groupe d'utilisateurs ou rôle), utilisez l'une des commandes suivantes :
   - était un objectif get-user-policy
   - était un objectif get-group-policy
   - était un objectif get-role-policy

3. Pour supprimer une politique en ligne d'une identité (utilisateur, groupe d'utilisateurs ou rôle qui n'est pas un rôle lié à un service), utilisez l'une des commandes suivantes :

   - était un objectif delete-user-policy
   - était un objectif delete-group-policy
   - était un objectif delete-role-policy

Ajouter des politiques IAM (AWS API)

Vous pouvez utiliser l' AWS API pour associer des politiques gérées qui contrôlent les autorisations ou spécifier une politique qui sert de limite d'autorisations. Vous pouvez également intégrer une politique en ligne.

Pour utiliser une politique gérée comme politique d'autorisation pour une entité (AWS API)

1. (Facultatif) Pour afficher des informations sur une politique, appelez les opérations suivantes :

   - Pour répertorier les politiques gérées : ListPolicies
   - Pour récupérer des informations détaillées sur une politique gérée, procédez comme suit : GetPolicy

2. Pour attacher une politique gérée à une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :

   - AttachUserPolicy
   - AttachGroupPolicy
   - AttachRolePolicy

Pour utiliser une politique gérée afin de définir une limite d'autorisations (AWS API)

1. (Facultatif) Pour afficher les informations d'une politique gérée, appelez les opérations suivantes :

   - Pour répertorier les politiques gérées : ListPolicies
   - Pour récupérer des informations détaillées sur une politique gérée, procédez comme suit : GetPolicy

2. Pour utiliser une politique gérée permettant de définir la limite d'autorisations pour une entité (utilisateur ou rôle), appelez l'une des opérations suivantes :

- PutUserPermissionsBoundary
- PutRolePermissionsBoundary

Pour intégrer une politique en ligne (API)AWS

Pour intégrer une politique en ligne à une identité (utilisateur, groupe d'utilisateurs ou rôle qui n'est pas un rôle lié à un service), appelez l'une des opérations suivantes :

- PutUserPolicy
- PutGroupPolicy
- PutRolePolicy

Supprimer les politiques IAM (AWS API)

Vous pouvez utiliser l' AWS API pour détacher les politiques gérées qui contrôlent les autorisations ou supprimer une politique qui sert de limite d'autorisations. Vous pouvez également supprimer une politique en ligne.

Pour détacher une politique gérée utilisée comme politique d'autorisation (AWS API)

1. (Facultatif) Pour afficher des informations sur une politique, appelez les opérations suivantes :
   - Pour répertorier les politiques gérées : ListPolicies
   - Pour récupérer des informations détaillées sur une politique gérée, procédez comme suit : GetPolicy
2. (Facultatif) Pour en savoir plus sur les relations entre les stratégies et les identités, appelez les opérations suivantes :
   - Pour répertorier les identités (utilisateurs IAM, groupes IAM et rôles IAM) auxquelles une politique gérée est attachée :
     - ListEntitiesForPolicy
   - Pour répertorier les politiques gérées attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies

3. Pour détacher une politique gérée d'une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :

   - DetachUserPolicy
   - DetachGroupPolicy
   - DetachRolePolicy

Pour supprimer une limite d'autorisation (AWS API)

1. (Facultatif)Pour afficher la politique gérée actuellement utilisée pour définir la limite d'autorisations d'un utilisateur ou rôle, appelez les opérations suivantes :

   - GetUser
   - GetRole
2. (Facultatif)Pour afficher les utilisateurs ou rôles sur lesquels une politique gérée est utilisée pour une limite d'autorisations, appelez l'opération suivante :

   - ListEntitiesForPolicy
3. (Facultatif) Pour afficher les informations d'une politique gérée, appelez les opérations suivantes :

   - Pour répertorier les politiques gérées : ListPolicies
   - Pour récupérer des informations détaillées sur une politique gérée, procédez comme suit : GetPolicy
4. Pour supprimer une limite d'autorisations d'un utilisateur ou d'un rôle, appelez l'une des opérations suivantes :

   - DeleteUserPermissionsBoundary
   - DeleteRolePermissionsBoundary

Pour supprimer une politique intégrée (AWS API)

1. (Facultatif) Pour répertorier toutes les politiques en ligne attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :

   - ListUserPolicies
   - ListGroupPolicies

- ListRolePolicies

2. (Facultatif) Pour récupérer un document de politique en ligne qui est intégré à une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :

   - GetUserPolicy
   - GetGroupPolicy
   - GetRolePolicy

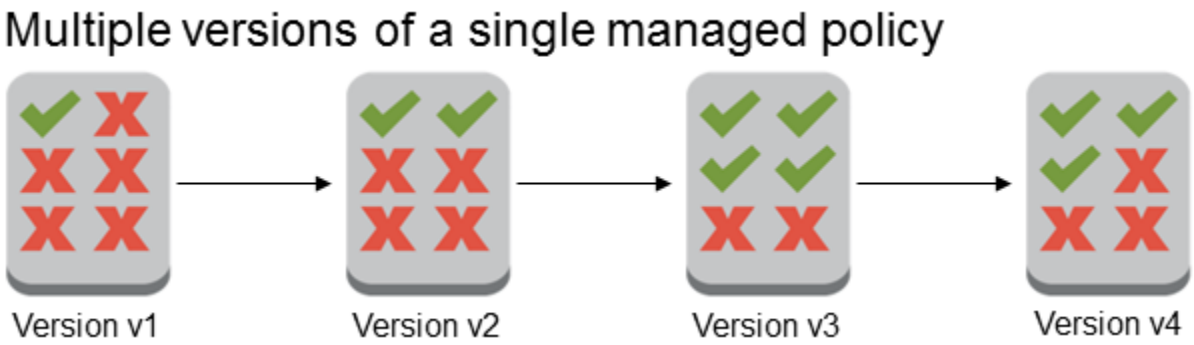
3. Pour supprimer une politique en ligne d'une identité (utilisateur, groupe d'utilisateurs ou rôle qui n'est pas un rôle lié à un service), appelez l'une des opérations suivantes :

   - DeleteUserPolicy
   - DeleteGroupPolicy
   - DeleteRolePolicy

Gestion des versions des politiques IAM

Lorsque vous apportez des modifications à une politique gérée par un client IAM ou à une politique AWS gérée, la stratégie modifiée ne remplace pas la politique existante. AWS À la place, IAM crée une nouvelle version de la politique gérée. IAM stocke jusqu'à cinq versions de vos politiques gérées par le client. IAM ne prend pas en charge la gestion des versions pour les politiques en ligne.

Le diagramme suivant illustre la gestion des versions pour une politique gérée par le client. Dans cet exemple, les versions de 1 à 4 sont enregistrées. Vous pouvez enregistrer jusqu'à cinq versions de politique gérées dans IAM. Lorsque vous créez une sixième version en modifiant une politique, vous pouvez choisir quelle version plus ancienne supprimer. Vous pouvez revenir à tout moment à l'une des quatre autres versions enregistrées.



Une version de politique est différente d'un élément de politique `Version`. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. Pour en savoir plus sur l'élément de politique `Version`, consultez Éléments de politique JSON IAM : Version.

Vous pouvez utiliser ces versions pour effectuer le suivi d'une politique gérée. Par exemple, vous pouvez modifier une politique gérée et découvrir ensuite que le changement a eu des effets inattendus. Dans ce cas, vous pouvez revenir à un version précédente de la politique gérée en définissant la version précédente comme version par défaut.

Les rubriques suivantes expliquent comment utiliser la gestion des versions avec les politiques gérées.

Rubriques

- Limites de version
- Utilisation des versions pour l'annulation de modifications
- Autorisations pour définir la version par défaut d'une politique
- Définition de la version par défaut des politiques gérées par le client

Limites de version

Une politique gérée peut avoir jusqu'à cinq versions. Si vous devez apporter des modifications à une politique gérée au-delà de cinq versions à partir de l' AWS Command Line Interface API ou de l' AWS API, vous devez d'abord supprimer une ou plusieurs versions existantes. Si vous utilisez le Console de gestion AWS, il n'est pas nécessaire de supprimer une version avant de modifier votre politique. Lorsque vous disposez d'une sixième version, une boîte de dialogue s'affiche pour vous inviter à supprimer une ou plusieurs versions personnalisées de votre politique. Vous pouvez également afficher le document de politique JSON pour vous aider à choisir. Pour plus d'informations sur cette boîte de dialogue, consultez la section the section called "Modification de politiques IAM".

Vous pouvez supprimer n'importe quelle version de la politique gérée de votre choix, sauf la version par défaut. Lorsque vous supprimez une version, les identificateurs de version des versions restantes ne changent pas. En conséquence, les identificateurs de version peuvent ne pas être séquentiels. Par exemple, si vous supprimez les versions v2 et v4 d'une politique gérée et que vous ajoutez deux nouvelles versions, les identificateurs de version restants peuvent être v1, v3, v5, v6 et v7.

Utilisation des versions pour l’annulation de modifications

Vous pouvez définir la version par défaut d'une politique gérée par le client pour annuler vos modifications. Par exemple, imaginez le scénario suivant:

Vous créez une politique gérée par le client qui autorise les utilisateurs à administrer un compartiment Amazon S3 spécifique à l'aide de la Console de gestion AWS. À sa création, votre politique gérée par le client dispose d'une version unique, identifiée comme v1. Cette version est donc définie automatiquement comme valeur par défaut. La politique fonctionne comme prévu.

Plus tard, vous mettrez à jour la politique pour ajouter l'autorisation d'administrer un second compartiment Amazon S3. IAM crée une nouvelle version de la politique, identifiée en tant que v2, qui contient vos modifications. Vous définissez la version v2 comme version par défaut, et peu de temps après, vos utilisateurs signalent qu'ils n'ont pas l'autorisation d'utiliser la console Amazon S3. Dans ce cas, vous pouvez revenir à la version v1 de la politique, que vous savez fonctionner comme prévu. Pour cela, vous définissez la version v1 comme version par défaut. Vous utilisateurs sont désormais en mesure d'utiliser la console Amazon S3 pour administrer le compartiment d'origine.

Plus tard, après avoir identifié l'erreur dans la version v2 de la politique, vous remettez à jour la politique pour ajouter l'autorisation d'administrer le second compartiment Amazon S3. IAM crée une autre version de la politique, identifiée en tant que v3. Vous définissez la version v3 comme version par défaut et cette version fonctionne comme prévu. À ce stade, vous supprimez la version v2 de la politique.

Autorisations pour définir la version par défaut d'une politique

Les autorisations qui sont requises pour définir la version par défaut d'une politique correspondent aux opérations d'API AWS pour la tâche. Vous pouvez utiliser les opérations d'API `CreatePolicyVersion` ou `SetDefaultPolicyVersion` pour définir la version par défaut d'une politique. Pour autoriser une personne à définir la version par défaut d'une politique existante, vous pouvez autoriser l'accès à l'action `iam:CreatePolicyVersion` ou l'action `iam:SetDefaultPolicyVersion`. L'action `iam:CreatePolicyVersion` permet de créer une nouvelle version de la politique et de définir cette version comme version par défaut. L'action `iam:SetDefaultPolicyVersion` permet de définir n'importe quelle version existante de la politique comme valeur par défaut.

> **Important**
>
> Pour empêcher un utilisateur d'apporter des modifications à la version par défaut d'une politique, vous devez lui refuser à la fois `iam:CreatePolicyVersion` et `iam:SetDefaultPolicyVersion`.

Vous pouvez utiliser la politique suivante pour interdire à un utilisateur de modifier une politique existante gérée par le client :

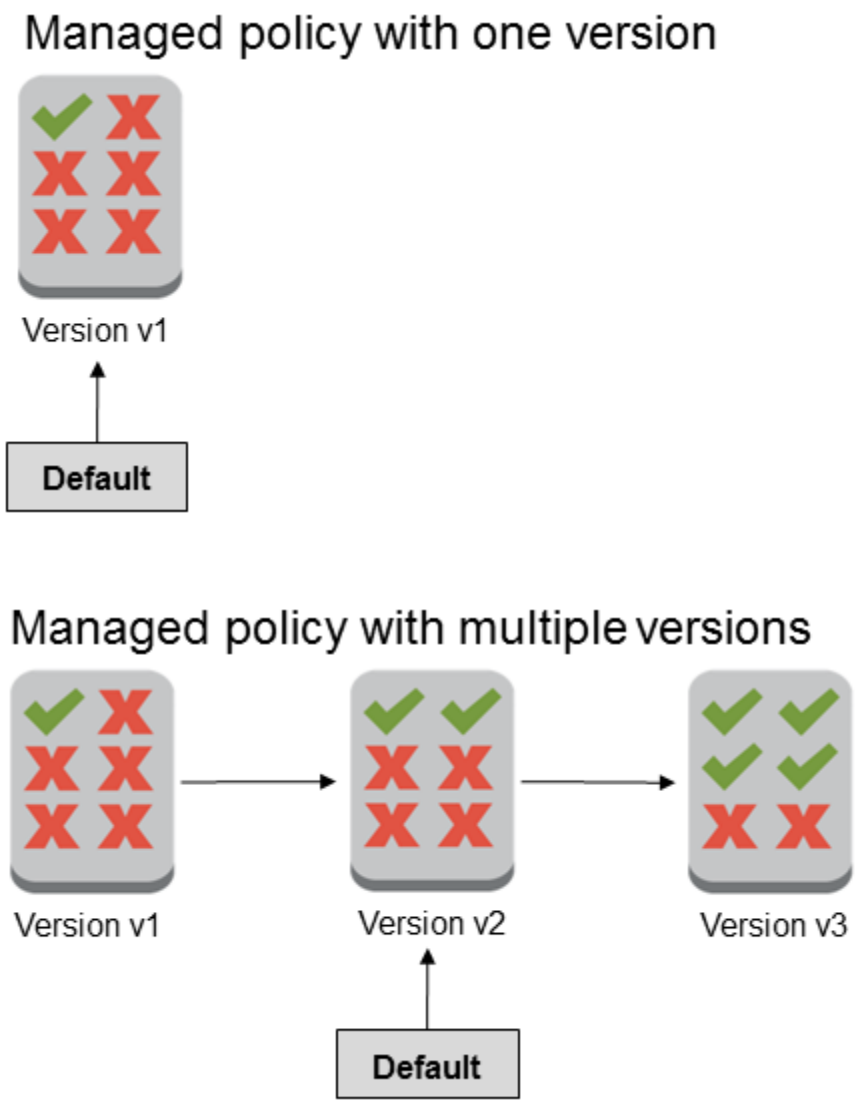
JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "iam:CreatePolicyVersion",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": "arn:aws:iam::*:policy/POLICY-NAME"
        }
    ]
}
```

Définition de la version par défaut des politiques gérées par le client

L'une des versions d'une politique gérée est définie comme la version par défaut. La version par défaut de la politique est la version d'exécution, c'est-à-dire qu'il s'agit de la version en vigueur pour toutes les entités de principal (utilisateurs IAM, groupes IAM et rôles IAM) auxquelles la politique gérée est attachée.

Lorsque vous créez une politique gérée par le client, la politique commence par une version unique identifiée comme v1. Pour les politiques gérées avec une version unique, cette version est définie automatiquement comme valeur par défaut. Pour les politiques gérées par le client avec plusieurs versions, vous choisissez la version à définir par défaut. Pour les politiques AWS gérées, la version par défaut est définie par AWS. Les diagrammes suivants illustrent ce concept.



Vous pouvez définir la version par défaut d'une politique gérée par le client pour appliquer cette version à chaque entité IAM (utilisateur, groupe d'utilisateurs et rôle) à laquelle la politique gérée est attachée. Vous ne pouvez pas définir la version par défaut d'une politique AWS gérée ou d'une politique intégrée.

Pour définir la version par défaut d'une politique gérée par le client (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de la stratégie à définir comme la version par défaut. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.

4. Choisissez l'onglet Versions de politique. Cochez la case située en regard de la version que vous souhaitez définir comme version par défaut, puis choisissez Définir par défaut.

Pour savoir comment définir la version par défaut d'une politique gérée par le client à partir de l'API AWS Command Line Interface ou de l' AWS API, consultezModification de politiques IAM (AWS CLI).

Modification de politiques IAM

Une politique est une entité qui, lorsqu'elle est attachée une identité ou à une ressource, définit les autorisations de cette dernière. Les politiques sont stockées AWS sous forme de documents JSON et sont attachées aux principes sous forme de politiques basées sur l'identité dans IAM. Vous pouvez attacher une politique basée sur identité à un principal (ou une identité), comme un groupe d'utilisateurs, un utilisateur ou un rôle IAM. Les politiques basées sur l'identité incluent les politiques AWS gérées, les politiques gérées par le client et les politiques intégrées. Vous pouvez modifier les politiques gérées par le client et les politiques intégrées dans IAM. AWS les politiques gérées ne peuvent pas être modifiées. Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

Il est généralement préférable d'utiliser des politiques gérées par le client plutôt que des politiques intégrées ou des politiques AWS gérées. AWS les politiques gérées fournissent généralement des autorisations administratives ou en lecture seule étendues. Les politiques en ligne ne peuvent pas être réutilisées avec d'autres identités ni être gérées en dehors de l'identité où elles existent. Pour plus de sécurité, octroyez le moindre privilège, c'est-à-dire n'accordez que les autorisations nécessaires à l'exécution de tâches spécifiques.

Lorsque vous créez ou modifiez des politiques IAM, vous AWS pouvez effectuer automatiquement la validation des politiques pour vous aider à créer une politique efficace avec le moindre privilège à l'esprit. Dans le Console de gestion AWS, IAM identifie les erreurs de syntaxe JSON, tandis qu'IAM Access Analyzer fournit des vérifications de politique supplémentaires avec des recommandations pour vous aider à affiner davantage vos politiques. Pour en savoir plus sur la validation de politiques, veuillez consulter Validation de politique IAM. Pour en savoir plus sur les vérifications des politiques IAM Access Analyzer et les recommandations exploitables, veuillez consulter Validation de politique IAM Access Analyzer.

Vous pouvez utiliser l' AWS API Console de gestion AWS AWS CLI, ou pour modifier les politiques gérées par le client et les politiques intégrées dans IAM. Pour plus d'informations sur l'utilisation de CloudFormation modèles pour ajouter ou mettre à jour des politiques, consultez la référence aux

types de Gestion des identités et des accès AWS ressources dans le Guide de CloudFormation l'utilisateur.

Rubriques

- Modification de politiques IAM (console)
- Modification de politiques IAM (AWS CLI)
- Modifier les politiques IAM (AWS API)

Modification de politiques IAM (console)

Une politique est une entité qui, lorsqu'elle est attachée une identité ou à une ressource, définit les autorisations de cette dernière. Vous pouvez utiliser le Console de gestion AWS pour modifier les politiques gérées par le client et les politiques intégrées dans IAM. AWS les politiques gérées ne peuvent pas être modifiées. Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

Pour plus d'informations sur la syntaxe et la structure de la politique, consultez les sections Politiques et autorisations dans Gestion des identités et des accès AWS et Référence de l'élément de politique JSON IAM.

Conditions préalables

Avant de modifier les autorisations d'une politique, vous devez passer en revue ses activités récentes au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Modification de politiques gérées par le client (console)

Vous pouvez modifier les politiques gérées par le client pour modifier les autorisations définies dans la politique depuis l' Console de gestion AWS. Une politique gérée le client peut avoir jusqu'à cinq versions. Ceci est important, car si vous devez modifier une politique gérée au-delà de cinq versions, l'interface Console de gestion AWS vous invite à choisir quelle version supprimer. Vous pouvez également modifier la version par défaut ou supprimer une version d'une politique avant de modifier celle-ci a pour éviter d'y être invité. Pour en savoir plus sur les versions, consultez Gestion des versions des politiques IAM.

Console

Pour modifier une politique gérée par le client

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des politiques, choisissez le nom de la politique à modifier. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Choisissez l'onglet Autorisations, puis Modifier.
5. Effectuez l'une des actions suivantes :

   - Choisissez l'option Visuel pour modifier votre politique sans comprendre la syntaxe JSON. Vous pouvez apporter des modifications au service, aux ressources d'actions ou à des conditions facultatives pour chaque bloc d'autorisation de votre politique. Vous pouvez également importer une politique pour ajouter des autorisations supplémentaires au bas de votre politique. Lorsque vous avez fini d'apporter des modifications, choisissez Suivant pour continuer.
   - Choisissez l'option JSON pour modifier votre politique en tapant ou en collant du texte dans la zone de texte JSON. Vous pouvez également importer une politique pour ajouter des autorisations supplémentaires au bas de votre politique. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant.

     > **Note**
     >
     > Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour de plus amples informations, veuillez consulter Restructuration de politique.

6. Sur la page Vérifier et enregistrer, vérifiez les Autorisations définies dans cette politique, puis choisissez Enregistrer les modifications pour enregistrer votre travail.
7. Si la politique gérée dispose déjà du maximum de cinq versions, une boîte de dialogue s'affiche lorsque vous choisissez Enregistrer les modifications. Pour enregistrer votre nouvelle version, la version la plus ancienne de la politique est supprimée et remplacée par

cette nouvelle version. Vous pouvez choisir de définir la nouvelle version en tant que version par défaut de la politique.

Choisissez Enregistrer les modifications pour enregistrer votre nouvelle version de la politique.

Définition de la version par défaut d’une politique gérée par le client (console)

Vous pouvez définir une version par défaut d'une politique gérée par le client à partir du Console de gestion AWS. Vous pouvez utiliser cette politique pour établir une configuration de base cohérente pour les autorisations au sein de votre organisation. Tous les nouveaux attachements de la politique utiliseront cet ensemble standardisé d’autorisations.

Console

Pour définir la version par défaut d'une politique gérée par le client (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de la stratégie à définir comme la version par défaut. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Choisissez l'onglet Versions de politique. Cochez la case située en regard de la version que vous souhaitez définir comme version par défaut, puis choisissez Définir par défaut.

Suppression d’une version de politique gérée par le client (console)

Vous devrez peut-être supprimer une version d’une politique gérée par le client afin de supprimer les autorisations obsolètes ou incorrectes qui ne sont plus nécessaires ou qui présentent des risques pour la sécurité. En ne conservant que les versions nécessaires, vous pouvez vous assurer de rester dans la limite des cinq versions de politiques gérées, ce qui vous laisse une marge de manœuvre pour les mises à jour et les améliorations futures. Vous pouvez supprimer une version d’une politique gérée par le client depuis l’ Console de gestion AWS.

Console

Pour supprimer une version d’une politique gérée par le client

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Choisissez le nom de la politique gérée par le client dont vous souhaitez supprimer une version. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Choisissez l'onglet Versions de politique. Activez la case à cocher en regard de la version à supprimer. Ensuite, choisissez Supprimer.
5. Confirmez que vous voulez supprimer la version, puis choisissez Supprimer.

Modification de politiques en ligne (console)

Vous devrez peut-être modifier une politique gérée par le client afin de mettre à jour ou d’affiner les autorisations accordées, en veillant à ce qu’elles restent conformes aux exigences de sécurité et aux besoins de contrôle d’accès en constante évolution de votre organisation. La modification vous permet d’ajuster le document JSON de la politique, en ajoutant, modifiant ou supprimant des actions, des ressources ou des conditions spécifiques afin de respecter le principe du moindre privilège et de vous adapter aux changements dans votre environnement ou vos processus. Vous pouvez modifier une politique en ligne à partir de l’interface Console de gestion AWS.

Console

Pour modifier une politique en ligne pour un utilisateur, un groupe d’utilisateurs ou un rôle

1. Dans le panneau de navigation, sélectionnez Users (Utilisateurs), User groups (Groupes d'utilisateurs) ou Roles (Rôles).
2. Choisissez le nom de l'utilisateur, du groupe d'utilisateurs ou du rôle ayant la politique que vous souhaitez modifier. Choisissez ensuite l'onglet Autorisations et développez la politique.
3. Pour modifier une politique en ligne, choisissez Modifier la politique.
4. Effectuez l’une des actions suivantes :

   - Choisissez l'option Visuel pour modifier votre politique sans comprendre la syntaxe JSON. Vous pouvez apporter des modifications au service, aux ressources d’actions ou à des conditions facultatives pour chaque bloc d’autorisation de votre politique. Vous pouvez

également importer une politique pour ajouter des autorisations supplémentaires au bas de votre politique. Lorsque vous avez fini d'apporter des modifications, choisissez Suivant pour continuer.

- Choisissez l'option JSON pour modifier votre politique en tapant ou en collant du texte dans la zone de texte JSON. Vous pouvez également importer une politique pour ajouter des autorisations supplémentaires au bas de votre politique. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant. Pour enregistrer vos modifications sans affecter les entités actuellement attachées, désactivez la case à cocher Définir comme version par défaut.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour de plus amples informations, veuillez consulter Restructuration de politique.

5. Sur la page Vérifier, vérifiez le résumé de la politique, puis choisissez Enregistrer les modifications pour enregistrer votre travail.

Modification de politiques IAM (AWS CLI)

Une politique est une entité qui, lorsqu'elle est attachée une identité ou à une ressource, définit les autorisations de cette dernière. Vous pouvez utiliser le AWS Command Line Interface (AWS CLI) pour modifier les politiques gérées par le client et les politiques intégrées dans IAM. AWS les politiques gérées ne peuvent pas être modifiées. Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

Pour plus d'informations sur la syntaxe et la structure de la politique, consultez les sections Politiques et autorisations dans Gestion des identités et des accès AWS et Référence de l'élément de politique JSON IAM.

Conditions préalables

Avant de modifier les autorisations d'une politique, vous devez passer en revue ses activités récentes au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Modification des politiques gérées par le client (AWS CLI)

Vous pouvez modifier une politique gérée par le client à partir du AWS CLI.

> **Note**
>
> Une politique gérée peut avoir jusqu'à cinq versions. Si vous devez modifier une politique gérée par le client au-delà de cinq versions, vous devez d'abord supprimer une ou plusieurs versions existantes.

Pour modifier une politique gérée par le client (AWS CLI)

1. (Facultatif) Pour afficher des informations sur une politique, exécutez les commandes suivantes :
   - Pour répertorier les stratégies gérées : list-policies
   - Pour récupérer les informations détaillées sur une politique gérée : get-policy
2. (Facultatif) Pour en savoir plus sur les relations entre les stratégies et les identités, exécutez les commandes suivantes :
   - Pour répertorier les identités (utilisateurs IAM, groupes IAM et rôles IAM) auxquelles une politique gérée est attachée :
     - list-entities-for-policy
   - Pour répertorier les politiques gérées attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle) :
     - list-attached-user-policies
     - list-attached-group-policies
     - list-attached-role-policies
3. Pour modifier une politique gérée par le client, exécutez la commande suivante :

- create-policy-version

4. (Facultatif) Pour valider une politique gérée par le client, exécutez la commande IAM Access Analyzer suivante :

   - validate-policy

Définition de la version par défaut d'une politique gérée par le client (AWS CLI)

Vous pouvez définir une version par défaut d'une politique gérée par le client à partir du AWS CLI.

Pour définir la version par défaut d'une politique gérée par le client (AWS CLI)

1. (Facultatif) Pour répertorier les stratégies gérées, exécutez la commande suivante :

   - list-policies

2. Pour définir la version par défaut d'une politique gérée par le client, exécutez la commande suivante :

   - set-default-policy-version

Suppression d'une version de politique gérée par le client (AWS CLI)

Vous pouvez supprimer une version d'une politique gérée par le client depuis l' AWS CLI.

Pour supprimer une version d'une politique gérée par le client (AWS CLI)

1. (Facultatif) Pour répertorier les stratégies gérées, exécutez la commande suivante :

   - list-policies

2. Pour supprimer une politique gérée par le client, exécutez la commande suivante :

   - delete-policy-version

Modification de politiques en ligne (AWS CLI)

Vous pouvez modifier une politique en ligne à partir de l'interface AWS CLI.

Pour modifier une politique en ligne (AWS CLI)

1. (Facultatif) Pour afficher des informations sur une politique, exécutez les commandes suivantes :
   - Pour répertorier les politiques en ligne attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle) :
     - list-user-policies
     - list-role-policies
     - list-group-policies
   - Pour récupérer les informations détaillées sur une politique en ligne :
     - get-user-policy
     - get-role-policy
     - get-group-policy
2. Pour modifier une politique en ligne, exécutez la commande suivante :
   - put-user-policy
   - put-role-policy
   - put-group-policy
3. (Facultatif) Pour valider une politique en ligne, exécutez la commande de l'analyseur d'accès IAM suivante :
   - validate-policy

Modifier les politiques IAM (AWS API)

Une politique est une entité qui, lorsqu'elle est attachée une identité ou à une ressource, définit les autorisations de cette dernière. Vous pouvez utiliser l' AWS API pour modifier les politiques gérées par le client et les politiques intégrées dans IAM. AWS les politiques gérées ne peuvent pas être modifiées. Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

Pour plus d'informations sur la syntaxe et la structure de la politique, consultez les sections Politiques et autorisations dans Gestion des identités et des accès AWS et Référence de l'élément de politique JSON IAM.

Conditions préalables

Avant de modifier les autorisations d'une politique, vous devez passer en revue ses activités récentes au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Modification des politiques gérées par le client (AWS API)

Vous pouvez modifier une politique gérée par le client à l'aide de l' AWS API.

> **Note**
>
> Une politique gérée peut avoir jusqu'à cinq versions. Si vous devez modifier une politique gérée par le client au-delà de cinq versions, vous devez d'abord supprimer une ou plusieurs versions existantes.

Pour modifier une politique gérée par le client (AWS API)

1. (Facultatif) Pour afficher des informations sur une politique, appelez les opérations suivantes :
   - Pour répertorier les politiques gérées : ListPolicies
   - Pour récupérer des informations détaillées sur une politique gérée : GetPolicy
2. (Facultatif) Pour en savoir plus sur les relations entre les stratégies et les identités, appelez les opérations suivantes :
   - Pour répertorier les identités (utilisateurs IAM, groupes IAM et rôles IAM) auxquelles une politique gérée est attachée :
     - ListEntitiesForPolicy
   - Pour répertorier les politiques gérées attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle) :
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies
3. Pour modifier une politique gérée par le client, appelez l'opération suivante :

- CreatePolicyVersion

4. (Facultatif) Pour valider une politique gérée par le client, appelez l'opération IAM Access Analyzer suivante :

   - ValidatePolicy

Configuration de la version par défaut d'une politique gérée par le client (AWS API)

Vous pouvez définir une version par défaut d'une politique gérée par le client à partir de l' AWS API.

Pour définir la version par défaut d'une politique gérée par le client (AWS API)

1. (Facultatif) Pour répertorier les stratégies gérées, appelez l'opération suivante :

   - ListPolicies

2. Pour définir la version par défaut d'une politique gérée par le client, appelez l'opération suivante :

   - SetDefaultPolicyVersion

Supprimer une version d'une politique gérée par le client (AWS API)

Vous pouvez supprimer une version d'une politique gérée par le client de l' AWS API.

Pour supprimer une version d'une politique gérée par le client (AWS API)

1. (Facultatif) Pour répertorier les stratégies gérées, appelez l'opération suivante :

   - ListPolicies

2. Pour supprimer une politique gérée par le client, appelez l'opération suivante :

   - DeletePolicyVersion

Modification des politiques intégrées (AWS API)

Vous pouvez modifier une politique en ligne depuis l' AWS API.

Pour modifier une politique intégrée (AWS API)

1. (Facultatif) Pour afficher les informations relatives à une politique intégrée, effectuez les opérations suivantes :
   - Pour répertorier les politiques en ligne attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle) :
     - ListUserPolicies
     - ListRolePolicies
     - ListGroupPolicies
   - Pour récupérer les informations détaillées sur une politique en ligne :
     - GetUserPolicy
     - GetRolePolicy
     - GetGroupPolicy
2. Pour modifier une politique intégrée, effectuez les opérations suivantes :
   - PutUserPolicy
   - PutRolePolicy
   - PutGroupPolicy
3. (Facultatif) Pour valider une politique en ligne, exécutez les opérations de l'analyseur d'accès IAM suivantes :
   - ValidatePolicy

Suppression de politiques IAM

Vous pouvez supprimer les politiques IAM à l'aide de Console de gestion AWS, du AWS Command Line Interface (AWS CLI) ou de l' AWS API.

> **Note**
>
> Toute suppression d'une politique IAM est définitive. Une fois la politique supprimée, elle ne peut plus être récupérée.

Pour plus d'informations sur la syntaxe et la structure de la politique IAM, consultez Politiques et autorisations dans Gestion des identités et des accès AWS et Référence de l'élément de politique JSON IAM.

Pour en savoir plus sur la différence entre les stratégies gérées et les stratégies en ligne, consultez Politiques gérées et politiques en ligne.

Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

Rubriques

- Suppression de politiques IAM (console)
- Suppression de politiques IAM (AWS CLI)
- Supprimer les politiques IAM (AWS API)

Suppression de politiques IAM (console)

Vous pouvez utiliser le Console de gestion AWS pour supprimer les politiques gérées par le client et les politiques intégrées dans IAM. Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

> **Note**
>
> Toute suppression d'une politique IAM est définitive. Une fois la politique supprimée, elle ne peut plus être récupérée.

Pour plus d'informations sur la syntaxe et la structure de la politique IAM, consultez Politiques et autorisations dans Gestion des identités et des accès AWS et Référence de l'élément de politique JSON IAM.

Pour en savoir plus sur la différence entre les stratégies gérées et les stratégies en ligne, consultez Politiques gérées et politiques en ligne.

Conditions préalables

Avant de supprimer une politique, vous devez passer en revue ses activités récentes au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal

(personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Création de politiques de rôle IAM (console)

Vous devrez peut-être supprimer une politique gérée par le client lorsqu'elle devient obsolète ou ne correspond plus aux exigences de sécurité et aux besoins de contrôle d'accès de votre organisation. En supprimant les politiques inutiles, vous réduisez les risques de sécurité potentiels associés aux politiques obsolètes ou non utilisées. Vous pouvez supprimer une politique gérée par le client pour la retirer de votre Compte AWS. Vous ne pouvez pas supprimer les politiques AWS gérées.

Console

Pour supprimer une politique gérée par le client

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Sélectionnez la case d'option en regard de la politique gérée par le client à supprimer. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Choisissez Actions, puis choisissez Supprimer.
5. Suivez les instructions pour confirmer la suppression de la politique, puis choisissez Supprimer.

Suppression de politiques en ligne (console)

Vous devrez peut-être supprimer une politique intégrée lorsque les autorisations spécifiques qu'elle accorde ne sont plus nécessaires pour l'utilisateur, le groupe ou le rôle IAM auquel elle est directement associée. La suppression des politiques en ligne inutiles permet de réduire le risque d'accès involontaire, d'autant plus que les politiques en ligne ne peuvent pas être réutilisées ou partagées entre plusieurs identités, contrairement aux politiques gérées. Vous pouvez supprimer une politique intégrée pour la supprimer de votre Compte AWS. Vous ne pouvez pas supprimer les politiques gérées AWS .

Console

Pour supprimer une politique intégrée pour un utilisateur, un groupe ou un rôle IAM

1. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), Users (Utilisateurs) ou Roles (Rôles).
2. Choisissez le nom du groupe d'utilisateurs, de l'utilisateur ou du rôle et la politique que vous souhaitez supprimer. Ensuite, choisissez l'onglet Autorisations.
3. Cochez les cases en regard des politiques à supprimer et choisissez Supprimer. Ensuite, dans la boîte de dialogue de confirmation, confirmez le retrait et la suppression de la politique.

   - Pour supprimer une politique en ligne dans Utilisateurs ou Rôles, choisissez Supprimer pour confirmer la suppression.
   - Si vous supprimez une seule politique en ligne dansGroupes d'utilisateurs, tapez le nom de la politique et choisissezSupprimer. Si vous supprimez plusieurs politiques en ligne dans User groups (Groupes d'utilisateurs), tapez le nombre de politiques à supprimer, suivi de **`inline policies`**, puis choisissez Delete (Supprimer). Par exemple, si vous supprimez trois politiques en ligne, tapez **`3 inline policies`**.

Suppression de politiques IAM (AWS CLI)

Vous pouvez utiliser le AWS Command Line Interface (AWS CLI) pour supprimer les politiques gérées par le client et les politiques intégrées dans IAM. Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

> **Note**
>
> Toute suppression d'une politique IAM est définitive. Une fois la politique supprimée, elle ne peut plus être récupérée.

Pour plus d'informations sur la syntaxe et la structure de la politique IAM, consultez Politiques et autorisations dans Gestion des identités et des accès AWS et Référence de l'élément de politique JSON IAM.

Pour en savoir plus sur la différence entre les stratégies gérées et les stratégies en ligne, consultez Politiques gérées et politiques en ligne.

Conditions préalables

Avant de supprimer une politique, vous devez passer en revue ses activités récentes au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Suppression de politiques gérées par le client (AWS CLI)

Vous pouvez supprimer une politique gérée par le client à partir de l'interface AWS Command Line Interface.

Pour supprimer une politique gérée par le client (AWS CLI)

1. (Facultatif) Pour afficher des informations sur une politique, exécutez les commandes suivantes :
   - Pour répertorier les stratégies gérées : list-policies
   - Pour récupérer les informations détaillées sur une politique gérée : get-policy
2. (Facultatif) Pour en savoir plus sur les relations entre les stratégies et les identités, exécutez les commandes suivantes :
   - Pour répertorier les identités (utilisateurs IAM, groupes IAM et rôles IAM) auxquelles une politique gérée est attachée, exécutez la commande suivante :
     - list-entities-for-policy
   - Pour répertorier les politiques gérées attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle), exécutez l'une des commandes suivantes :
     - list-attached-user-policies
     - list-attached-group-policies
     - list-attached-role-policies
3. Pour supprimer une politique gérée par le client, exécutez la commande suivante :
   - delete-policy

Suppression de politiques en ligne (AWS CLI)

Vous pouvez supprimer une politique en ligne à partir de l' AWS CLI.

Pour supprimer une politique en ligne (AWS CLI)

1. (Facultatif) Pour répertorier toutes les politiques en ligne attachées à une identité (utilisateur, groupe d'utilisateurs, rôle), utilisez l'une des commandes suivantes :
   - était un objectif list-user-policies
   - était un objectif list-group-policies
   - était un objectif list-role-policies
2. (Facultatif) Pour récupérer un document de politique en ligne qui est intégré à une identité (utilisateur, groupe d'utilisateurs ou rôle), utilisez l'une des commandes suivantes :
   - était un objectif get-user-policy
   - était un objectif get-group-policy
   - était un objectif get-role-policy
3. Pour supprimer une politique en ligne d'une identité (utilisateur, groupe d'utilisateurs ou rôle qui n'est pas un rôle lié à un service), utilisez l'une des commandes suivantes :
   - était un objectif delete-user-policy
   - était un objectif delete-group-policy
   - était un objectif delete-role-policy

Supprimer les politiques IAM (AWS API)

Vous pouvez utiliser l' AWS API pour supprimer les politiques gérées par le client et les politiques intégrées dans IAM. Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

> **Note**
>
> Toute suppression d'une politique IAM est définitive. Une fois la politique supprimée, elle ne peut plus être récupérée.

Pour plus d'informations sur la syntaxe et la structure de la politique IAM, consultez Politiques et autorisations dans Gestion des identités et des accès AWS et Référence de l'élément de politique JSON IAM.

Pour en savoir plus sur la différence entre les stratégies gérées et les stratégies en ligne, consultez Politiques gérées et politiques en ligne.

Conditions préalables

Avant de supprimer une politique, vous devez passer en revue ses activités récentes au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Supprimer les politiques gérées par le client (AWS API)

Vous pouvez supprimer une politique gérée par le client à l'aide de l' AWS API.

Pour supprimer une politique gérée par le client (AWS API)

1. (Facultatif) Pour afficher des informations sur une politique, appelez les opérations suivantes :
   - Pour répertorier les politiques gérées : ListPolicies
   - Pour récupérer des informations détaillées sur une politique gérée, procédez comme suit : GetPolicy
2. (Facultatif) Pour en savoir plus sur les relations entre les stratégies et les identités, appelez les opérations suivantes :
   - Pour répertorier les identités (utilisateurs IAM, groupes IAM et rôles IAM) auxquelles une politique gérée est attachée, appelez l'opération suivante :
     - ListEntitiesForPolicy
   - Pour répertorier les politiques gérées attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies
3. Pour supprimer une politique gérée par le client, appelez l'opération suivante :

- DeletePolicy

Suppression de politiques intégrées (AWS API)

Vous pouvez supprimer une politique intégrée à l'aide de l' AWS API.

Pour supprimer une politique intégrée (AWS API)

1. (Facultatif) Pour répertorier toutes les politiques en ligne attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :
   - ListUserPolicies
   - ListGroupPolicies
   - ListRolePolicies
2. (Facultatif) Pour récupérer un document de politique en ligne qui est intégré à une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :
   - GetUserPolicy
   - GetGroupPolicy
   - GetRolePolicy
3. Pour supprimer une politique en ligne d'une identité (utilisateur, groupe d'utilisateurs ou rôle qui n'est pas un rôle lié à un service), appelez l'une des opérations suivantes :
   - DeleteUserPolicy
   - DeleteGroupPolicy
   - DeleteRolePolicy

Affiner les autorisations en AWS utilisant les dernières informations consultées

En tant qu'administrateur, vous pouvez accorder davantage d'autorisations aux ressources IAM (rôles, utilisateurs, groupes d'utilisateurs ou politiques) qu'elles n'en nécessitent. IAM fournit les dernières informations consultées pour vous aider à identifier les autorisations inutilisées et les supprimer. Vous pouvez utiliser les dernières informations consultées pour affiner vos politiques et n'autoriser l'accès qu'aux services et actions utilisés par vos identités et politiques IAM. Cela vous

permet de mieux respecter la bonne pratique que l'on appelle principe du moindre privilège. Vous pouvez afficher les dernières informations consultées pour les identités ou les politiques qui existent dans IAM ou AWS Organizations.

Vous pouvez surveiller en permanence les dernières informations consultées à l'aide d'analyseurs d'accès non utilisés. Pour plus d'informations, consultez Résultats des accès externes et non utilisés.

Rubriques

- Types des dernières informations consultées pour IAM
- Dernières informations consultées pour AWS Organizations
- Choses à savoir sur les dernières informations consultées
- Autorisations nécessaires
- Résoudre les problèmes liés à l'IAM et aux entités AWS Organizations
- Où AWS suit les dernières informations consultées
- Affichage des dernières informations consultées pour IAM
- Afficher les dernières informations consultées pour AWS Organizations
- Exemples de scénarios d'utilisation des dernières informations consultées
- Services et actions concernant les dernières informations consultées relatives à une action IAM

Types des dernières informations consultées pour IAM

Vous pouvez afficher deux types de dernières informations consultées pour les identités IAM : les informations sur les services AWS autorisés et les informations sur les actions autorisées. Les informations incluent la date et l'heure auxquelles la tentative d'accès à une AWS API a été faite. Pour les actions, les dernières informations consultées font état des actions de gestion des services. Les actions de gestion comprennent les actions de création, de suppression et de modification. Pour en savoir plus sur la façon d'afficher les dernières informations consultées pour IAM, consultez Affichage des dernières informations consultées pour IAM.

Pour obtenir des exemples de scénarios concernant l'utilisation des dernières informations consultées dans le but de prendre des décisions sur les autorisations que vous accordez à vos identités IAM, veuillez consulter Exemples de scénarios d'utilisation des dernières informations consultées.

Pour en savoir plus sur la façon dont les informations relatives aux actions de gestion sont fournies, consultez Choses à savoir sur les dernières informations consultées.

Dernières informations consultées pour AWS Organizations

Si vous vous connectez à l'aide des informations d'identification du compte de gestion, vous pouvez consulter les informations du dernier accès au service pour une AWS Organizations entité ou une politique de votre organisation. AWS Organizations les entités incluent la racine de l'organisation, les unités organisationnelles (OUs) ou les comptes. Les dernières informations consultées pour AWS Organizations incluent des informations sur les services autorisés par une politique de contrôle des services (SCP). Les informations indiquent quels principaux (utilisateur root, utilisateur IAM ou fonction) d'une organisation ou d'un compte ont tenté pour la dernière fois d'accéder au service et à quel moment. Pour en savoir plus sur le rapport et sur la façon de consulter les dernières informations consultées pour AWS Organizations, voirAfficher les dernières informations consultées pour AWS Organizations.

Pour des exemples de scénarios d'utilisation des dernières informations consultées pour prendre des décisions concernant les autorisations que vous accordez à vos AWS Organizations entités, voirExemples de scénarios d'utilisation des dernières informations consultées.

Choses à savoir sur les dernières informations consultées

Avant d'utiliser les dernières informations d'un rapport auxquelles vous avez accédé pour modifier les autorisations d'une identité ou d'une AWS Organizations entité IAM, passez en revue les informations suivantes.

- Période de suivi : l'activité récente apparaît dans la console IAM dans un délai de quatre heures. La période de suivi pour les informations de service s'étend sur au moins 400 jours, en fonction de la date à laquelle le service a commencé à suivre les informations sur les actions. La période de suivi des information sur les actions Amazon S3 a commencé le 12 avril 2020. La période de suivi des actions Amazon EC2, IAM et Lambda a commencé le 7 avril 2021. La période de suivi pour tous les autres services a débuté le 23 mai 2023. Pour obtenir la liste des services pour lesquels les dernières informations consultées relatives à une action sont disponibles, veuillez consulter Services et actions concernant les dernières informations consultées relatives à une action IAM. Pour plus d'informations concernant les régions dans lesquelles les dernières informations consultées relatives à une action sont disponibles, veuillez consulter Où AWS suit les dernières informations consultées.
- Tentatives signalées — Les dernières données consultées par le service incluent toutes les tentatives d'accès à une AWS API, et pas seulement les tentatives réussies. Cela inclut toutes les tentatives effectuées à l' Console de gestion AWS aide de l' AWS API via l'un ou SDKs l'autre des outils de ligne de commande. Une entrée inattendue dans les données relatives aux services

consultés en dernier ne signifie pas que votre compte a été mis en danger, car la demande a pu être refusée. Référez-vous à vos CloudTrail journaux en tant que source officielle pour obtenir des informations sur tous les appels d'API et savoir s'ils ont été réussis ou non.

- PassRole— L'`iam:PassRole`action n'est pas suivie et n'est pas incluse dans les dernières informations consultées sur l'action IAM.
- Dernières informations consultées relatives à une action : les dernières informations consultées relatives à une action sont disponibles pour les actions de gestion de service auxquelles des identités IAM ont accédé. Veuillez consulter la liste des services et de leurs actions pour lesquels les derniers accès relatifs à une action ont fourni des informations.

> **Note**
>
> Les dernières informations consultées relatives à une action ne sont pas disponibles pour aucun événement de plan de données.

- Événements de gestion : IAM fournit des informations d'action pour les événements de gestion des services qui sont enregistrés par CloudTrail. Parfois, les événements de gestion CloudTrail sont également appelés opérations de plan de contrôle ou événements de plan de contrôle. Les événements de gestion fournissent une visibilité sur les opérations administratives effectuées sur les ressources de votre entreprise Compte AWS. Pour en savoir plus sur les événements de gestion dans CloudTrail, consultez la section Journalisation des événements de gestion dans le Guide de AWS CloudTrail l'utilisateur.
- Report owner (Propriétaire du rapport) : seul le principal qui génère un rapport peut afficher les détails de ce dernier. Cela signifie que lorsque vous consultez les informations contenues dans le Console de gestion AWS, vous devrez peut-être attendre qu'elles soient générées et chargées. Si vous utilisez l' AWS API AWS CLI or pour obtenir les détails du rapport, vos informations d'identification doivent correspondre à celles du principal qui a généré le rapport. Si vous utilisez des informations d'identification temporaires pour un rôle ou un utilisateur principal AWS STS fédéré, vous devez générer et récupérer le rapport au cours de la même session. Pour plus d'informations sur les principaux de session à rôle endossé, reportez-vous à la section AWS Éléments de politique JSON : Principal.
- Ressources IAM : les dernières informations consultées concernant IAM incluent les ressources IAM (rôles, utilisateurs, groupes IAM et politiques) de votre compte. Les dernières informations consultées pour AWS Organizations incluent les principaux (utilisateurs IAM, rôles IAM ou les Utilisateur racine d'un compte AWS) dans l'entité spécifiée. AWS Organizations Les tentatives non authentifiées sont exclues des dernières informations consultées.

- Types de politique IAM : les dernières informations consultées concernant IAM incluent les services qui sont autorisés par les politiques d'une identité IAM. Il s'agit de politiques attachées à un rôle ou attachées à un utilisateur directement ou via un groupe. L'accès autorisé par d'autres types de politique n'est pas inclus dans votre rapport. Les types de politiques exclus incluent les politiques basées sur les ressources, les listes de contrôle d'accès AWS Organizations SCPs, les limites d'autorisations IAM et les politiques de session. Les autorisations fournies par les rôles liés au service sont définies par le service auquel ils sont liés et ne peuvent pas être modifiées dans IAM. Pour en savoir plus sur les rôles liés au service, consultez Créer un rôle lié à un service Pour en savoir plus sur les différents types de politique sont évalués pour autoriser ou refuser l'accès, consultez Logique d'évaluation de politiques.
- AWS Organizations types de politiques — Les informations pour AWS Organizations incluent uniquement les services autorisés par les politiques de contrôle des services héritées d'une AWS Organizations entité (SCPs). SCPs sont des politiques associées à une racine, une unité d'organisation ou un compte. L'accès autorisé par d'autres types de politique n'est pas inclus dans votre rapport. Les types de politique exclus comprennent les politiques basées sur une identité, les politiques basées sur les ressources, les listes de contrôle d'accès, les limites d'autorisations IAM et les politiques de sessions. Pour en savoir plus sur les différents types de politique qui sont évaluées pour autoriser ou refuser l'accès, veuillez consulter Logique d'évaluation de politiques.
- Spécification d'un ID de politique — Lorsque vous utilisez l' AWS API AWS CLI or pour générer un rapport sur les dernières informations consultées dans AWS Organizations, vous pouvez éventuellement spécifier un ID de politique. Le rapport inclut des informations pour les services qui sont uniquement autorisées par cette politique. Les informations incluent l'activité la plus récente du compte dans l' AWS Organizations entité spécifiée ou les enfants de l'entité. Pour plus d'informations, consultez aws iam generate-organizations-access-report or GenerateOrganizationsAccessReport.
- AWS Organizations compte de gestion : vous devez vous connecter au compte de gestion de votre organisation pour consulter les dernières informations du service auxquelles vous avez accédé. Vous pouvez choisir d'afficher les informations relatives au compte de gestion à l'aide de la console IAM, de l' AWS CLI API ou de l' AWS API. Le rapport qui en résulte répertorie tous les AWS services, car le compte de gestion n'est pas limité par SCPs. Si vous spécifiez un ID de politique dans l'interface de ligne de commande (CLI) ou l'API, la politique est ignorée. Pour chaque service, le rapport inclut uniquement les informations du compte de gestion. Toutefois, les rapports relatifs aux autres AWS Organizations entités ne renvoient pas d'informations sur l'activité du compte de gestion.
- AWS Organizations paramètres : un administrateur doit l'activer SCPs à la racine de votre organisation pour que vous puissiez générer des données pour AWS Organizations.

Autorisations nécessaires

Pour afficher les dernières informations consultées dans le Console de gestion AWS, vous devez disposer d'une politique qui accorde les autorisations nécessaires.

Autorisations pour des informations IAM

Pour utiliser la console IAM pour afficher les dernières informations consultées pour un utilisateur, un rôle ou une politique IAM, vous devez posséder une politique qui inclut les actions suivantes :

- `iam:GenerateServiceLastAccessedDetails`
- `iam:Get*`
- `iam:List*`

Ces autorisations permettent à l'utilisateur de voir les éléments suivants :

- Les utilisateurs, les groupes ou les rôles attachés à une politique gérée
- Les services auxquels un utilisateur ou un rôle peut accéder
- La dernière fois où ils se sont connectés au service
- La dernière fois qu'ils ont tenté d'utiliser une action Amazon EC2, IAM, Lambda ou Amazon S3 spécifique

Pour utiliser l' AWS API AWS CLI or afin de consulter les dernières informations consultées pour IAM, vous devez disposer d'autorisations correspondant à l'opération que vous souhaitez utiliser :

- `iam:GenerateServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetailsWithEntities`
- `iam:ListPoliciesGrantingServiceAccess`

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'affichage des dernières informations IAM consultées. En outre, il permet un accès en lecture seule à tous les éléments IAM. Cette politique définit des autorisations pour l'accès à la console et par programmation.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateServiceLastAccessedDetails",
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    }
}
```

Autorisations d' AWS Organizations information

Pour utiliser la console IAM pour afficher un rapport pour les entités racine, unité organisationnelle ou compte dans AWS Organizations, vous devez avoir une politique qui inclut les actions suivantes :

- `iam:GenerateOrganizationsAccessReport`
- `iam:GetOrganizationsAccessReport`
- `organizations:DescribeAccount`
- `organizations:DescribeOrganization`
- `organizations:DescribeOrganizationalUnit`
- `organizations:DescribePolicy`
- `organizations:ListChildren`
- `organizations:ListParents`
- `organizations:ListPoliciesForTarget`
- `organizations:ListRoots`
- `organizations:ListTargetsForPolicy`

Pour utiliser l' AWS API AWS CLI or afin de consulter les dernières informations du service auquel vous avez accédé AWS Organizations, vous devez disposer d'une politique incluant les actions suivantes :

- `iam:GenerateOrganizationsAccessReport`
- `iam:GetOrganizationsAccessReport`
- `organizations:DescribePolicy`
- `organizations:ListChildren`
- `organizations:ListParents`
- `organizations:ListPoliciesForTarget`
- `organizations:ListRoots`
- `organizations:ListTargetsForPolicy`

Cet exemple montre comment créer une politique basée sur l'identité qui permet d'afficher les informations du dernier accès au service pour. AWS Organizations De plus, il permet un accès en lecture seule à l'ensemble. AWS Organizations Cette politique définit des autorisations pour l'accès à la console et par programmation.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateOrganizationsAccessReport",
            "iam:GetOrganizationsAccessReport",
            "organizations:Describe*",
            "organizations:List*"
        ],
        "Resource": "*"
    }
}
```

Vous pouvez également utiliser la clé de OrganizationsPolicyId condition iam : pour autoriser la génération d'un rapport uniquement pour une AWS Organizations politique spécifique. Pour un exemple de politique, consultez IAM : Afficher les informations du dernier accès au service pour une politique AWS Organizations.

Résoudre les problèmes liés à l'IAM et aux entités AWS Organizations

Dans certains cas, le Console de gestion AWS dernier tableau d'informations auquel vous avez accédé est peut-être vide. Ou peut-être que votre demande AWS CLI ou celle de AWS l'API renvoie un ensemble d'informations vide ou un champ nul. Dans ces cas, passez en revue les problèmes suivants :

- Pour les dernières informations consultées relatives à une action, cette dernière, que vous vous attendez à voir, peut ne pas être renvoyée dans la liste. Cela peut se produire soit parce que l'identité IAM n'est pas autorisée à effectuer l'action, soit parce AWS qu'elle n'assure pas encore le suivi de l'action pour les dernières informations consultées.
- Pour un utilisateur IAM, assurez-vous qu'il possède au moins une politique gérée ou une politique en ligne attachée, directement ou via l'appartenance à des groupes.
- Pour un groupe IAM, vérifiez qu'il possède au moins une politique gérée ou une politique en ligne attachée.
- Pour un groupe IAM, le rapport renvoie uniquement les dernières informations consultées relatives au service pour les membres ayant utilisé les politiques du groupe pour accéder à un service. Pour savoir si un membre a utilisé d'autres politiques, vérifiez les dernières informations consultées pour cet utilisateur.
- Pour un rôle IAM, vérifiez que le rôle possède au moins une politique gérée ou une politique en ligne attachée.
- Pour une entité IAM (utilisateur ou rôle), recherchez les autres types de politique qui peuvent affecter les autorisations de cette entité. Il s'agit notamment des politiques basées sur les ressources, des listes de contrôle d'accès, des AWS Organizations politiques, des limites d'autorisations IAM ou des politiques de session. Pour plus d’informations, consultez Types de politique ou Évaluation des politiques pour les demandes au sein d’un même compte.
- Dans le cas d'une politique IAM, assurez-vous que la politique gérée spécifiée est attachée à au moins un utilisateur, un groupe avec des membres ou un rôle.
- Pour une AWS Organizations entité (root, UO ou compte), assurez-vous que vous êtes connecté à l'aide AWS Organizations des informations d'identification du compte de gestion.
- Vérifiez qu'SCPs ils sont activés à la racine de votre organisation.
- Les dernières informations consultées relatives à une action ne sont disponibles que pour les actions répertoriées dans Services et actions concernant les dernières informations consultées relatives à une action IAM.

Lorsque vous apportez des modifications, attendez au moins 4 heures avant que l'activité ne s'affiche dans votre console IAM. Si vous utilisez l' AWS API AWS CLI or, vous devez générer un nouveau rapport pour afficher les informations mises à jour.

Où AWS suit les dernières informations consultées

AWS collecte les dernières informations consultées pour les AWS régions standard. Lorsque AWS vous ajoutez des régions supplémentaires, celles-ci sont ajoutées au tableau suivant, y compris la date de AWS début du suivi des informations dans chaque région.

- Informations sur le service : la période de suivi des services correspond au moins à 400 jours, ou moins si votre région a commencé à suivre cette fonctionnalité au cours des 400 derniers jours.
- Information sur les actions : la période de suivi des actions de gestion Amazon S3 a commencé le 12 avril 2020. La période de suivi des actions de gestion Amazon EC2, IAM et Lambda a commencé le 7 avril 2021. La période de suivi des actions de gestion pour tous les autres services a débuté le 23 mai 2023. Si la date de suivi d'une région est postérieure au 23 mai 2023, les dernières informations consultées relatives à une action dans cette région débuteront à la date la plus tardive.

Nom de la région	Région	Date de début de suivi
USA Est (Ohio)	us-east-2	27 octobre 2017
USA Est (Virginie du Nord)	us-east-1	1er octobre 2015
USA Ouest (Californie du Nord)	us-west-1	1er octobre 2015
USA Ouest (Oregon)	us-west-2	1er octobre 2015
Afrique (Le Cap)	af-south-1	22 avril 2020
Asie-Pacifique (Hong Kong)	ap-east-1	24 avril 2019
Asie-Pacifique (Hyderabad)	ap-south-2	22 novembre 2022
Asie-Pacifique (Jakarta)	ap-southeast-3	13 décembre 2021
Asie-Pacifique (Melbourne)	ap-southeast-4	23 janvier 2023

Nom de la région	Région	Date de début de suivi
Asie-Pacifique (Mumbai)	ap-south-1	27 juin 2016
Asie-Pacifique (Osaka)	ap-northeast-3	11 février 2018
Asie-Pacifique (Séoul)	ap-northeast-2	6 janvier 2016
Asie-Pacifique (Singapour)	ap-southeast-1	1er octobre 2015
Asie-Pacifique (Sydney)	ap-southeast-2	1er octobre 2015
Asie-Pacifique (Tokyo)	ap-northeast-1	1er octobre 2015
Canada (Centre)	ca-central-1	28 octobre 2017
Europe (Francfort)	eu-central-1	1er octobre 2015
Europe (Irlande)	eu-west-1	1er octobre 2015
Europe (Londres)	eu-west-2	28 octobre 2017
Europe (Milan)	eu-south-1	28 avril 2020
Europe (Paris)	eu-west-3	18 décembre 2017
Europe (Espagne)	eu-south-2	15 novembre 2022
Europe (Stockholm)	eu-north-1	12 décembre 2018
Europe (Zurich)	eu-central-2	8 novembre 2022
Israël (Tel Aviv)	il-central-1	1er août 2023
Moyen-Orient (Bahreïn)	me-south-1	29 juillet 2019
Moyen-Orient (EAU)	me-central-1	30 août 2022
Amérique du Sud (São Paulo)	sa-east-1	11 décembre 2015
AWS GovCloud (USA Est)	us-gov-east-1	1er juillet 2023

Nom de la région	Région	Date de début de suivi
AWS GovCloud (US-Ouest)	us-gov-west-1	1er juillet 2023

Si une région n'est pas répertoriée dans le tableau précédent, cette région ne fournit pas encore les dernières informations consultées.

Une AWS région est un ensemble de AWS ressources dans une zone géographique. Les régions sont regroupées en partitions. Les régions standard sont les celles qui appartiennent à la partition `aws`. Pour plus d'informations sur les différentes partitions, consultez Amazon Resource Names (ARNs) Format dans le Références générales AWS. Pour plus d'informations sur les régions, voir également À propos AWS des régions dans le Références générales AWS.

Affichage des dernières informations consultées pour IAM

Vous pouvez consulter les dernières informations consultées pour IAM à l'aide de l' AWS API Console de gestion AWS AWS CLI, ou. Veuillez consulter la liste des services et de leurs actions pour lesquels les dernières informations consultées sont affichées. Pour de plus amples informations sur les dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Vous pouvez consulter les informations relatives aux types de ressources suivants dans IAM. Dans chaque cas, les informations incluent les services autorisés pour la période donnée :

- Utilisateur : afficher la dernière fois que l'utilisateur a tenté d'accéder à chaque service autorisé.
- User group (Groupe) : consultez les informations sur la dernière fois où un membre du groupe a tenté d'accéder à chaque service autorisé. Ce rapport inclut également le nombre total de membres qui ont tenté un accès.
- Role (Rôle) : affichez la dernière fois où quelqu'un a utilisé le rôle pour tenter d'accéder à chaque service autorisé.
- Policy (Politique) : consultez les informations sur la dernière fois où un utilisateur ou un rôle a tenté d'accéder à chaque service autorisé. Ce rapport inclut également le nombre total d'entités qui ont tenté un accès.

> **Note**
>
> Avant de consulter les informations d'accès d'une ressource IAM, vous devez bien comprendre la période couverte par le rapport, les entités rapportées et les types de politique évalués pour vos informations. Pour en savoir plus, consultez the section called "Choses à savoir sur les dernières informations consultées".

Affichage des informations pour IAM (console)

Vous pouvez afficher les dernières informations consultées pour IAM dans l'onglet Dernier accès de la console IAM.

Pour afficher les informations pour IAM (console)

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), Users (Utilisateurs), Roles (Rôles) ou Policies (Politiques).
3. Choisissez un nom d'utilisateur, de groupe d'utilisateurs, de rôle ou de politique pour ouvrir la page Récapitulatif et sélectionnez l'onglet Dernier accès. Affichez les informations suivantes, en fonction de la ressource que vous avez choisie :

   - User group (Groupe d'utilisateurs) : affiche la liste des services auxquels les membres du groupe d'utilisateurs peuvent accéder. Vous pouvez également afficher la dernière fois qu'un membre a accédé au service, quelles politiques de groupe d'utilisateurs il a utilisées et quel membre du groupe d'utilisateurs a effectué la demande. Choisissez le nom de la politique pour savoir s'il s'agit d'une politique gérée ou d'une politique de groupe d'utilisateurs en ligne. Choisissez le nom du membre du groupe pour afficher tous les membres du groupe d'utilisateurs et quand ils ont consulté pour la dernière fois le service.
   - User (Utilisateur) : affiche la liste des services auxquels l'utilisateur peut accéder. Vous pouvez également afficher la date à laquelle ils ont accédé au service pour la dernière fois et les politiques qui sont actuellement associées à l'utilisateur. Choisissez le nom de la politique pour savoir s'il s'agit d'une politique gérée, d'une politique d'utilisateur en ligne ou d'une politique en ligne pour le groupe.
   - Role (Rôle) : affichez la liste des services auxquels le rôle peut accéder, à quel moment le rôle a accédé au service pour la dernière fois et quelles politiques ont été utilisées. Choisissez le

nom de la politique pour savoir s'il s'agit d'une politique gérée ou d'une politique de rôle en ligne.

- Policy (Politique) : affiche la liste des services avec des actions autorisées dans la politique. Vous pouvez également afficher la dernière fois où la politique a été utilisée pour accéder au service et quelle entité (utilisateur ou rôle) l'a utilisé. La date Dernier accès inclut également le moment où l'accès est accordé à cette politique par l'intermédiaire d'une autre politique. Choisissez le nom de l'entité pour connaître les entités ayant cette politique attachée et quand elles ont consulté pour la dernière fois le service.

4. Dans la colonne Service du tableau, choisissez le nom de l'un des services qui inclut les dernières informations consultées relatives à une action pour afficher une liste des actions de gestion auxquelles les entités IAM ont tenté d'accéder. Vous pouvez afficher l' Région AWS et un horodatage indiquant la dernière fois que quelqu'un a tenté d'exécuter l'action.
5. La colonne Dernier accès s'affiche pour les services et les actions de gestion des services qui incluent les dernières informations consultées relatives à une action. Examinez les résultats possibles suivants qui sont renvoyés dans cette colonne. Ces résultats varient selon qu'un service ou une action est autorisé, qu'il a été consulté et qu'il est suivi ou non par AWS les dernières informations consultées.

   il y a <nombre de> jours

   Nombre de jours écoulés depuis que le service ou l'action a été utilisé pendant la période de suivi. La période de suivi des services s'étend sur les 400 derniers jours. La période de suivi des actions Amazon S3 a commencé le 12 avril 2020. La période de suivi des actions Amazon EC2, IAM et Lambda a commencé le 7 avril 2021. La période de suivi pour tous les autres services a débuté le 23 mai 2023. Pour en savoir plus sur les dates de début du suivi pour chacune d'entre elles Région AWS, consultezOù AWS suit les dernières informations consultées.

   Aucun accès pendant la période de suivi

   Le service ou l'action suivi n'a pas été utilisé par une entité au cours de la période de suivi.

   Vous pouvez disposez d'autorisations pour une action qui n'apparaît pas dans la liste. C'est le cas lorsque les informations de suivi de l'action ne sont pas actuellement prises en compte par AWS. Vous ne devez pas prendre de décisions sur les autorisations uniquement en raison de l'absence d'informations de suivi. Nous vous recommandons plutôt d'utiliser ces informations

pour éclairer et fonder votre stratégie globale d'octroi du moindre privilège. Vérifiez vos politiques pour confirmer que le niveau d'accès est approprié.

Affichage des informations pour IAM (AWS CLI)

Vous pouvez utiliser le AWS CLI pour récupérer des informations sur la dernière fois qu'une ressource IAM a été utilisée pour tenter d'accéder à des AWS services et à des actions Amazon S3, Amazon EC2, IAM et Lambda. Une ressource IAM peut être un utilisateur, un groupe d'utilisateurs, un rôle ou une politique.

Pour afficher les informations pour IAM (AWS CLI)

1. Générez un rapport. La demande doit inclure l'ARN de la ressource IAM (utilisateur, groupe d'utilisateurs, rôle ou politique) pour lequel vous voulez un rapport. Vous pouvez spécifier le niveau de granularité que vous souhaitez générer dans le rapport pour afficher les détails sur l'accès pour les services ou pour les services et les actions. La demande renvoie un `job-id` que vous pouvez ensuite utiliser dans les opérations `get-service-last-accessed-details-with-entities` et `get-service-last-accessed-details` pour surveiller l'état `job-status` jusqu'à ce que la tâche soit terminée.

   - aws iam -détails generate-service-last-accessed
2. Récupérez les informations sur le rapport à l'aide du paramètre `job-id` de l'étape précédente.

   - aws iam -détails get-service-last-accessed

   Cette opération renvoie les informations suivantes, en fonction du type de ressource et du niveau de granularité que vous avez demandé dans l'opération `generate-service-last-accessed-details` :

   - User (Utilisateur) : renvoie une liste de services auxquels l'utilisateur spécifié peut accéder. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative de l'utilisateur, ainsi que l'ARN de l'utilisateur.
   - Groupe d'utilisateurs : renvoie la liste des services auxquels les membres du groupe d'utilisateurs spécifié peuvent accéder à l'aide des politiques attachées au groupe d'utilisateurs. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative effectuée par un membre du groupe d'utilisateurs. Il renvoie également l'ARN de cet utilisateur et le nombre total de membres du groupe d'utilisateurs qui ont tenté d'accéder au service.

Utilisez cette GetServiceLastAccessedDetailsWithEntitiesopération pour récupérer la liste de tous les membres.

- Role (Rôle) : renvoie la liste des services auxquels le rôle spécifié peut accéder. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative du rôle, ainsi que l'ARN du rôle.
- Policy (Politique) : renvoie la liste des services pour lesquels la politique spécifiée autorise l'accès. Pour chaque service, l'opération renvoie la date et l'heure auxquelles une entité (utilisateur ou rôle) a tenté pour la dernière fois d'accéder au service à l'aide de la politique. Elle renvoie également l'ARN de cette entité et le nombre total d'entités ayant tenté l'accès.

3. En savoir plus sur les entités qui ont utilisé les autorisations de groupe d'utilisateurs ou de politique lors de la tentative d'accès à un service spécifique. Cette opération renvoie la liste des entités avec l'ARN, l'ID, le nom, le chemin, le type (utilisateur ou rôle) de chaque entité, et la date à laquelle elles ont tenté pour la dernière fois d'accéder au service. Vous pouvez également utiliser cette opération pour les utilisateurs et les rôles, mais elle ne renvoie que les informations relatives à cette entité.

   - c'est moi - get-service-last-accessed details-with-entities

4. En savoir plus sur les politiques basées sur une identité qu'une identité (utilisateur, groupe d'utilisateurs ou rôle) a utilisées lors de la tentative d'accès à un service spécifique. Lorsque vous spécifiez une identité et un service, cette opération renvoie une liste de stratégies d'autorisation que l'identité peut utiliser pour accéder au service spécifié. Cette opération donne l'état actuel des stratégies et ne dépend pas du rapport généré. Elle ne retourne pas non plus d'autres types de politiques, tels que les politiques basées sur les ressources, les listes de contrôle d'accès, les politiques AWS Organizations , les limites d'autorisations IAM ou les politiques de sessions. Pour plus d'informations, consultez Types de politique ou Évaluation des politiques pour les demandes au sein d'un même compte.

   - aws iam -access list-policies-granting-service

Affichage des informations pour IAM (AWS API)

Vous pouvez utiliser l' AWS API pour récupérer des informations sur la dernière fois qu'une ressource IAM a été utilisée pour tenter d'accéder à des AWS services et à des actions Amazon S3, Amazon EC2, IAM et Lambda. Une ressource IAM peut être un utilisateur, un groupe d'utilisateurs, un rôle ou une politique. Vous pouvez spécifier le niveau de granularité à générer dans le rapport pour afficher les détails des services ou des services et des actions.

Pour consulter les informations relatives à l'IAM (AWS API)

1. Générez un rapport. La demande doit inclure l'ARN de la ressource IAM (utilisateur, groupe d'utilisateurs, rôle ou politique) pour lequel vous voulez un rapport. Elle renvoie un `JobId` que vous pouvez ensuite utiliser dans les opérations `GetServiceLastAccessedDetailsWithEntities` et `GetServiceLastAccessedDetails` pour surveiller l'état `JobStatus` jusqu'à ce que la tâche soit terminée.

   - GenerateServiceLastAccessedDetails

2. Récupérez les informations sur le rapport à l'aide du paramètre `JobId` de l'étape précédente.

   - GetServiceLastAccessedDetails

   Cette opération renvoie les informations suivantes, en fonction du type de ressource et du niveau de granularité que vous avez demandé dans l'opération `GenerateServiceLastAccessedDetails` :

   - User (Utilisateur) : renvoie une liste de services auxquels l'utilisateur spécifié peut accéder. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative de l'utilisateur, ainsi que l'ARN de l'utilisateur.
   - Groupe d'utilisateurs : renvoie la liste des services auxquels les membres du groupe d'utilisateurs spécifié peuvent accéder à l'aide des politiques attachées au groupe d'utilisateurs. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative effectuée par un membre du groupe d'utilisateurs. Il renvoie également l'ARN de cet utilisateur et le nombre total de membres du groupe d'utilisateurs qui ont tenté d'accéder au service. Utilisez cette GetServiceLastAccessedDetailsWithEntitiesopération pour récupérer la liste de tous les membres.
   - Role (Rôle) : renvoie la liste des services auxquels le rôle spécifié peut accéder. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative du rôle, ainsi que l'ARN du rôle.
   - Policy (Politique) : renvoie la liste des services pour lesquels la politique spécifiée autorise l'accès. Pour chaque service, l'opération renvoie la date et l'heure auxquelles une entité (utilisateur ou rôle) a tenté pour la dernière fois d'accéder au service à l'aide de la politique. Elle renvoie également l'ARN de cette entité et le nombre total d'entités ayant tenté l'accès.

3. En savoir plus sur les entités qui ont utilisé les autorisations de groupe d'utilisateurs ou de politique lors de la tentative d'accès à un service spécifique. Cette opération renvoie la liste des entités avec l'ARN, l'ID, le nom, le chemin, le type (utilisateur ou rôle) de chaque entité, et la date à laquelle elles ont tenté pour la dernière fois d'accéder au service. Vous pouvez également utiliser cette opération pour les utilisateurs et les rôles, mais elle ne renvoie que les informations relatives à cette entité.

   - GetServiceLastAccessedDetailsWithEntities

4. En savoir plus sur les politiques basées sur une identité qu'une identité (utilisateur, groupe d'utilisateurs ou rôle) a utilisées lors de la tentative d'accès à un service spécifique. Lorsque vous spécifiez une identité et un service, cette opération renvoie une liste de stratégies d'autorisation que l'identité peut utiliser pour accéder au service spécifié. Cette opération donne l'état actuel des stratégies et ne dépend pas du rapport généré. Elle ne retourne pas non plus d'autres types de politiques, tels que les politiques basées sur les ressources, les listes de contrôle d'accès, les politiques AWS Organizations , les limites d'autorisations IAM ou les politiques de sessions. Pour plus d’informations, consultez Types de politique ou Évaluation des politiques pour les demandes au sein d’un même compte.

   - ListPoliciesGrantingServiceAccess

Afficher les dernières informations consultées pour AWS Organizations

Vous pouvez consulter les informations relatives au dernier accès au service pour AWS Organizations utiliser la console IAM ou AWS CLI l' AWS API. Pour connaître les informations importantes sur les données, les autorisations requises, la résolution de problèmes et les régions prises en charge, veuillez consulter Affiner les autorisations en AWS utilisant les dernières informations consultées.

Lorsque vous vous connectez à la console IAM à l'aide AWS Organizations des informations d'identification du compte de gestion, vous pouvez consulter les informations relatives à n'importe quelle entité de votre organisation. AWS Organizations les entités incluent la racine de l'organisation, les unités organisationnelles (UO) et les comptes. Vous pouvez également utiliser la console IAM pour afficher des informations relatives aux politiques de contrôle des services (SCP) de votre organisation. IAM affiche une liste des services qui sont autorisés par chacune des SCP qui s'appliquent à l'entité. Pour chaque service, vous pouvez consulter les informations les plus récentes sur l'activité du compte de l' AWS Organizations entité sélectionnée ou de ses enfants.

Lorsque vous utilisez l' AWS API AWS CLI or avec les informations d'identification du compte de gestion, vous pouvez générer un rapport pour toutes les entités ou politiques de votre organisation. Un rapport de données par programmation pour une entité comprend une liste des services qui sont autorisés par les stratégies de contrôle de service qui s'appliquent à l'entité. Pour chaque service, le rapport inclut l'activité la plus récente pour les comptes de l'entité AWS Organizations spécifiée ou de la sous-arborescence de l'entité.

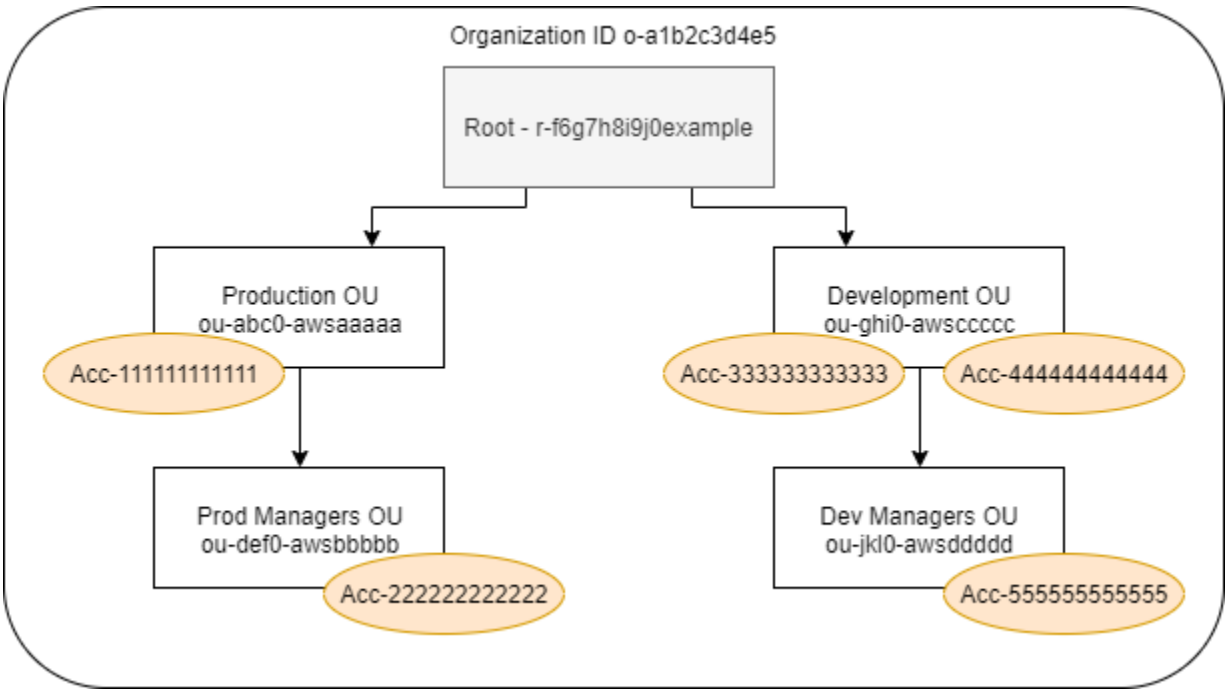
Lorsque vous générez un rapport programmatique pour une politique, vous devez spécifier une AWS Organizations entité. Ce rapport inclut une liste des services qui sont autorisés par la stratégie de contrôle de service spécifiée. Pour chaque service, il inclut la dernière activité du compte dans l'entité ou dans les enfants de l'entité qui sont autorisés par cette politique. Pour plus d'informations, consultez aws iam generate-organizations-access-report ou. GenerateOrganizationsAccessReport

Avant d'afficher le rapport, assurez-vous de comprendre les exigences et les informations du compte de gestion, la période couverte par le rapport, les entités concernées par le rapport, ainsi que les types de politiques évaluées. Pour en savoir plus, consultez the section called “Choses à savoir sur les dernières informations consultées”.

Comprenez les AWS Organizations chemin de l'entité

Lorsque vous utilisez l' AWS API AWS CLI or pour générer un rapport d' AWS Organizations accès, vous devez spécifier un chemin d'entité. Un chemin est une représentation textuelle de la structure d'une entité AWS Organizations .

Vous pouvez créer un chemin d'entité à l'aide de la structure connue de votre organisation. Supposons, par exemple, que vous disposez de la structure organisationnelle suivante dans AWS Organizations.



Le chemin de l'unité d'organisation Dev Managers est créé à l'aide des ID de l'organisation, de la racine et de toutes les unités d'organisation du chemin jusqu'à l'unité d'organisation incluse.

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-ghi0-awsccccc/ou-jkl0-awsdddddd/
```

Le chemin du compte dans l'unité d'organisation Production est généré à l'aide des ID de l'organisation, de la racine, de l'unité d'organisation et du numéro de compte.

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-abc0-awsaaaaa/111111111111/
```

> **Note**
>
> Les ID d'organisation sont globalement uniques, mais les ID d'unité d'organisation et les ID racine ne sont uniques qu'au sein d'une organisation. Cela signifie qu'aucune organisation ne partage le même ID d'organisation. Toutefois, une autre organisation peut avoir le même ID d'unité d'organisation ou ID racine que vous. Nous vous recommandons de toujours inclure l'ID d'organisation lorsque vous spécifiez une unité d'organisation ou une racine.

Affichage des informations pour AWS Organizations (console)

Vous pouvez utiliser la console IAM pour afficher les informations relatives aux derniers services consultés pour votre compte racine, une unité organisationnelle ou une politique.

Pour afficher les informations relatives à la racine (console)

1. Connectez-vous à l' Console de gestion AWS aide AWS Organizations des informations d'identification du compte de gestion et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation situé sous la section Access reports (Rapports d'accès), choisissez Organization activity (Activité de l'organisation).
3. Dans la page Organization activity (Activité de l'organisation), choisissez Root (Racine).
4. Sur l'onglet Details and activity (Détails et activité), affichez la section Service access report (Rapport d'accès aux services). Les informations comprennent une liste des services autorisés par les politiques qui sont attachées directement à la racine. Les informations vous indiquent également le compte à partir duquel le service a été consulté pour la dernière fois et le moment auquel cette consultation a eu lieu. Pour de plus amples informations sur le principal ayant consulté le service, connectez-vous en tant qu'administrateur à ce compte et affichez les dernières informations consultées relatives au service IAM.
5. Sélectionnez l'onglet Attached SCPs (SCP attachés) pour afficher la liste des politiques de contrôle des services (SCP) qui sont attachées à la racine. IAM vous indique le nombre d'entités cibles auxquelles chaque politique est attachée. Vous pouvez utiliser ces informations pour déterminer la stratégie de contrôle de service à passer en revue.
6. Choisissez le nom d'une politique de contrôle de service pour afficher tous les services qu'elle autorise. Pour chaque service, affichez le compte à partir duquel le service a été consulté pour la dernière fois et le moment auquel cette consultation a eu lieu.
7. Choisissez Modifier dans AWS Organizations pour afficher des informations supplémentaires et modifier le SCP dans la AWS Organizations console. Pour plus d'information, consultez la section relative aux mises à jour Over-the-Air dans le AWS Organizations User Guide.

Pour afficher les informations d'une unité d'organisation ou d'un compte (console)

1. Connectez-vous à l' Console de gestion AWS aide AWS Organizations des informations d'identification du compte de gestion et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.

2. Dans le panneau de navigation situé sous la section Access reports (Rapports d'accès), choisissez Organization activity (Activité de l'organisation).
3. Sur la page Organization activity (Activité de l'organisation), développez la structure de votre organisation. Ensuite, choisissez le nom de l'unité organisationnelle ou du compte que vous souhaitez afficher (à l'exception du compte de gestion).
4. Sur l'onglet Details and activity (Détails et activité), affichez la section Service access report (Rapport d'accès aux services). Les informations comprennent une liste des services qui sont autorisés par les stratégies de contrôle de service attachées à l'unité organisationnelle ou au compte et à tous ses parents. Les informations vous indiquent également le compte à partir duquel le service a été consulté pour la dernière fois et le moment auquel cette consultation a eu lieu. Pour de plus amples informations sur le principal ayant consulté le service, connectez-vous en tant qu'administrateur à ce compte et affichez les dernières informations consultées relatives au service IAM.
5. Sélectionnez l'onglet Attached SCPs (SCP attachés) pour afficher la liste des politiques de contrôle des services (SCP) qui sont attachées directement à l'UO ou au compte. IAM vous indique le nombre d'entités cibles auxquelles chaque politique est attachée. Vous pouvez utiliser ces informations pour déterminer la stratégie de contrôle de service à passer en revue.
6. Choisissez le nom d'une politique de contrôle de service pour afficher tous les services qu'elle autorise. Pour chaque service, affichez le compte à partir duquel le service a été consulté pour la dernière fois et le moment auquel cette consultation a eu lieu.
7. Choisissez Modifier dans AWS Organizations pour afficher des informations supplémentaires et modifier le SCP dans la AWS Organizations console. Pour plus d'information, consultez la section relative aux mises à jour Over-the-Air dans le AWS Organizations User Guide.

Pour afficher les informations du compte de gestion (console)

1. Connectez-vous à l' Console de gestion AWS aide AWS Organizations des informations d'identification du compte de gestion et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation situé sous la section Access reports (Rapports d'accès), choisissez Organization activity (Activité de l'organisation).
3. Sur la page Organization activity (Activité de l'organisation), développez la structure de votre organisation et choisissez le nom de votre compte de gestion.
4. Sur l'onglet Details and activity (Détails et activité), affichez la section Service access report (Rapport d'accès aux services). Les informations incluent une liste de tous les services AWS .

Le compte de gestion n'est pas limité par les SCP. Les informations vous indiquent si le compte a consulté le service dernièrement et le moment auquel cette consultation s'est produite. Pour de plus amples informations sur le principal ayant consulté le service, connectez-vous en tant qu'administrateur à ce compte et affichez les dernières informations consultées relatives au service IAM.

5. Sélectionnez l'onglet Attached SCPs (Stratégies de contrôle des services attachées) pour vérifier qu'il n'y a pas de stratégies de contrôle de service attachées, car le compte est le compte de gestion.

Pour afficher les informations d'une politique (console)

1. Connectez-vous à l' Console de gestion AWS aide AWS Organizations des informations d'identification du compte de gestion et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation situé sous la section Access reports (Rapports d'accès), choisissez Service control policies (SCPs) (Politiques de contrôle des services (SCP)).
3. Dans la page Service control policies (SCPs) (Politiques de contrôle de service (SCP)), affichez la liste des politiques de votre organisation. Vous pouvez consulter le nombre d'entités cibles auxquelles chaque politique est attachée.
4. Choisissez le nom d'une politique de contrôle de service pour afficher tous les services qu'elle autorise. Pour chaque service, affichez le compte à partir duquel le service a été consulté pour la dernière fois et le moment auquel cette consultation a eu lieu.
5. Choisissez Modifier dans AWS Organizations pour afficher des informations supplémentaires et modifier le SCP dans la AWS Organizations console. Pour plus d'information, consultez Mise à jour d'une stratégie de contrôle de service (SCP) dans le Guide de l'utilisateur AWS Organizations .

Affichage des informations pour AWS Organizations (AWS CLI)

Vous pouvez l'utiliser AWS CLI pour récupérer les informations sur les derniers accès au service pour votre AWS Organizations root, votre unité d'organisation, votre compte ou votre politique.

Pour visionner AWS Organizations informations sur le dernier accès au service (AWS CLI)

1. Utilisez les informations d'identification de votre compte de AWS Organizations gestion avec l'IAM et AWS Organizations les autorisations requises, et confirmez que les SCP sont activés

pour votre root. Pour de plus amples informations, veuillez consulter Choses à savoir sur les dernières informations consultées.

2. Générez un rapport. La demande doit inclure le chemin de l' AWS Organizations entité (racine, unité d'organisation ou compte) pour laquelle vous souhaitez obtenir un rapport. Si vous le souhaitez, vous pouvez inclure un paramètre `organization-policy-id` pour afficher un rapport pour une politique spécifique. Cette commande renvoie un élément `job-id` que vous pouvez ensuite utiliser dans la commande `get-organizations-access-report` pour surveiller l'élément `job-status` jusqu'à ce que la tâche soit terminée.

   - aws iam generate-organizations-access-report

3. Récupérez les informations sur le rapport à l'aide du paramètre `job-id` de l'étape précédente.

   - aws iam get-organizations-access-report

   Cette commande renvoie une liste des services auxquels les membres de l'entité peuvent accéder. Pour chaque service, la commande renvoie la date et l'heure de la dernière tentative d'un membre du compte et le chemin d'entité du compte. Elle renvoie également le nombre total de services qui sont disponibles en accès et le nombre de services qui n'ont pas été consultés. Si vous avez spécifié le paramètre facultatif `organizations-policy-id`, les services qui sont disponibles en accès sont ceux qui sont autorisés par la politique spécifiée.

Affichage des informations pour AWS Organizations (AWS API)

Vous pouvez utiliser l' AWS API pour récupérer les informations relatives au dernier accès au service pour votre AWS Organizations root, votre unité d'organisation, votre compte ou votre politique.

Pour visionner AWS Organizations informations sur le dernier accès au service (AWS API)

1. Utilisez les informations d'identification de votre compte de AWS Organizations gestion avec l'IAM et AWS Organizations les autorisations requises, et confirmez que les SCP sont activés pour votre root. Pour de plus amples informations, veuillez consulter Choses à savoir sur les dernières informations consultées.
2. Générez un rapport. La demande doit inclure le chemin de l' AWS Organizations entité (racine, unité d'organisation ou compte) pour laquelle vous souhaitez obtenir un rapport. Si vous le souhaitez, vous pouvez inclure un paramètre `OrganizationsPolicyId` pour afficher un rapport pour une politique spécifique. Cette opération renvoie un élément `JobId` que vous

pouvez ensuite utiliser dans l'opération `GetOrganizationsAccessReport` pour surveiller l'état `JobStatus` jusqu'à ce que la tâche soit terminée.

- GenerateOrganizationsAccessReport

3. Récupérez les informations sur le rapport à l'aide du paramètre `JobId` de l'étape précédente.

   - GetOrganizationsAccessReport

   Cette opération renvoie une liste des services auxquels les membres de l'entité peuvent accéder. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative d'un membre du compte et le chemin d'entité du compte. Elle renvoie également le nombre total de services qui sont disponibles en accès et le nombre de services qui n'ont pas été consultés. Si vous avez spécifié le paramètre facultatif `OrganizationsPolicyId`, les services qui sont disponibles en accès sont ceux qui sont autorisés par la politique spécifiée.

Exemples de scénarios d'utilisation des dernières informations consultées

Vous pouvez utiliser les dernières informations consultées pour prendre des décisions concernant les autorisations que vous accordez à vos entités ou AWS Organizations entités IAM. Pour de plus amples informations, veuillez consulter Affiner les autorisations en AWS utilisant les dernières informations consultées.

> **Note**
>
> Avant de consulter les informations d'accès d'une entité ou d'une politique dans IAM AWS Organizations, assurez-vous de bien comprendre la période de reporting, les entités déclarées et les types de politiques évalués pour vos données. Pour en savoir plus, consultez the section called "Choses à savoir sur les dernières informations consultées".

En tant qu'administrateur, il vous appartient d'équilibrer l'accessibilité et le principe du moindre privilège requis pour votre entreprise.

Utilisation des informations pour réduire les autorisations d'un groupe IAM

Vous pouvez utiliser les dernières informations consultés pour réduire les autorisations de groupe IAM et inclure uniquement les services dont vos utilisateurs ont besoin. Cette méthode est une étape importante dans l'attribution du moindre privilège au niveau service.

Par exemple, Paulo Santos est l'administrateur chargé de définir les autorisations des AWS utilisateurs pour Example Corp. Cette société vient de commencer à utiliser AWS, et l'équipe de développement du logiciel n'a pas encore défini les AWS services qu'elle utilisera. Paulo souhaite accorder à l'équipe l'autorisation d'accéder uniquement aux services dont elle a besoin, mais comme cela n'est pas encore défini, il leur attribue temporairement les autorisations utilisateur. Ensuite, il utilise les dernières informations consultées pour réduire les autorisations du groupe.

Paulo crée une politique gérée nommée `ExampleDevelopment` à l'aide du texte JSON suivant. Puis, il l'attache à un groupe appelé `Development` et ajoute tous les développeurs au groupe.

> **Note**
>
> Les utilisateurs avancés de Paulo peuvent avoir besoin d'autorisations `iam:CreateServiceLinkedRole` pour utiliser certains services et fonctionnalités. Il comprend que l'ajout de cette autorisation permet aux utilisateurs de créer n'importe quel rôle lié au service. Il accepte ce risque pour ses utilisateurs avancés.

JSON

```
{

    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessToAllServicesExceptPeopleManagement",
            "Effect": "Allow",
            "NotAction": [
                "iam:*",
                "organizations:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

Paulo décide d'attendre 90 jours avant d'afficher les dernières informations consultées pour le groupe `Development` à l'aide de la Console de gestion AWS. Il affiche la liste des services que les membres du groupe ont consultés. Il apprend que les utilisateurs ont accédé à cinq services la semaine dernière : AWS CloudTrail Amazon CloudWatch Logs, Amazon EC2 et Amazon S3. AWS KMS Ils ont eu accès à quelques autres services lors de leur première évaluation AWS, mais plus depuis.

Paulo décide de réduire les autorisations de politique pour n'inclure que ces cinq services, ainsi que l'IAM et les AWS Organizations actions requises. Il modifie la politique `ExampleDevelopment` à l'aide du texte JSON suivant.

> **Note**
>
> Les utilisateurs avancés de Paulo peuvent avoir besoin d'autorisations `iam:CreateServiceLinkedRole` pour utiliser certains services et fonctionnalités. Il comprend que l'ajout de cette autorisation permet aux utilisateurs de créer n'importe quel rôle lié au service. Il accepte ce risque pour ses utilisateurs avancés.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessToListedServices",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "kms:*",
                "cloudtrail:*",
                "logs:*",
                "ec2:*"
```

```
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
}
```

Pour réduire davantage les autorisations, Paulo peut consulter les événements du compte dans AWS CloudTrail Event history (Historique des événements). Là, il peut afficher les informations détaillées des événements qu'il peut utiliser pour réduire les autorisations de la politique et inclure uniquement les actions et les ressources dont les développeurs ont besoin. Pour plus d'informations, consultez la section Affichage CloudTrail des événements dans la CloudTrail console dans le Guide de AWS CloudTrail l'utilisateur.

Utilisation des informations pour réduire les autorisations d'un utilisateur IAM

Vous pouvez utiliser les dernières informations consultées pour réduire les autorisations d'un utilisateur IAM individuel.

Par exemple, Martha Rivera est une administratrice informatique chargée de s'assurer que les employés de son entreprise ne disposent pas d'autorisations excessives AWS . Dans le cadre d'un contrôle de sécurité régulier, elle passe en revue les autorisations de tous les utilisateurs IAM. L'un des ces utilisateurs est un développeur d'applications nommé Nikhil Jayashankar, qui occupait précédemment le rôle d'ingénieur sécurité. Dans la mesure où ses tâches ont évolué, Nikhil est à la fois membre du groupe `app-dev` et du groupe `security-team`. Le groupe `app-dev` pour son nouveau poste octroie des autorisations à plusieurs services, notamment Amazon EC2, Amazon EBS, Auto Scaling, Amazon S3, Route 53 et Elastic Transcoder. Le `security-team` groupe correspondant à son ancien poste accorde des autorisations à IAM et CloudTrail.

En tant qu'administrateur, Martha se connecte à la console IAM, sélectionnez Utilisateurs, le nom `nikhilj`, puis l'onglet Dernier accès.

Martha passe en revue la colonne Dernier accès et remarque que Nikhil n'a pas récemment accédé à IAM, Route 53 CloudTrail, Amazon Elastic Transcoder et à un certain nombre d'autres services. AWS Nikhil a accédé à Amazon S3. Martha choisit S3 dans la liste des services et apprend que Nikhil a effectué quelques actions Amazon S3 `List` au cours des deux dernières semaines. Au sein de son entreprise, Martha confirme que Nikhil n'a plus besoin d'accéder à IAM pour des raisons professionnelles et qu'il ne fait CloudTrail plus partie de l'équipe de sécurité interne.

Martha est maintenant prête à agir sur le service et des dernières information consultées relatives à l'action. Toutefois, à la différence du groupe de l'exemple précédent, un utilisateur IAM comme `nikhilj` peut être soumis à plusieurs politiques et appartenir à plusieurs groupes. Martha doit procéder avec précaution pour éviter de modifier par inadvertance l'accès de `nikhilj` ou d'autres membres du groupe. En plus d'apprendre quel accès Nikhil doit avoir, elle doit déterminer comment ces autorisations lui sont accordées.

Martha choisit l'onglet Autorisations, où elle consulte les politiques attachées directement à `nikhilj` et celles attachées à partir d'un groupe. Elle développe chaque politique et affiche le récapitulatif de la politique pour savoir quelle politique autorise l'accès aux services que Nikhil n'utilise pas :

- IAM — La politique `IAMFullAccess` AWS gérée est directement attachée au `security-team` groupe `nikhilj` et y est attachée.
- CloudTrail — La politique `AWS CloudTrailReadOnlyAccess` AWS gérée est attachée au `security-team` groupe.
- Route 53 : la politique `App-Dev-Route53` gérée par le client est attachée au groupe `app-dev`.
- Elastic Transcoder : la politique `App-Dev-ElasticTranscoder` gérée par le client est attachée au groupe `app-dev`.

Martha décide de supprimer la politique `IAMFullAccess` AWS gérée qui est directement associée à`nikhilj`. Elle supprime également l'appartenance de Nikhil au groupe `security-team`. Ces deux actions suppriment l'accès inutile à IAM et CloudTrail.

Les autorisations de Nikhil d'accéder à Route 53 et Elastic Transcoder sont octroyées par le groupe `app-dev`. Bien que Nikhil n'utilise pas ces services, il se peut que d'autres membres le fassent. Martha examine les dernières informations consultées pour le groupe `app-dev` et apprend que plusieurs membres ont récemment accédé à Route 53 et Amazon S3. Mais aucun membre du groupe n'a accédé à Elastic Transcoder au cours de la dernière année. Supprime du groupe la politique `App-Dev-ElasticTranscoder` gérée par le client.

Martha passe ensuite en revue les dernières informations consultées pour la politique `App-Dev-ElasticTranscoder` gérée par le client. Elle apprend que la politique n'est pas attachée à d'autres identités IAM. Elle vérifie au sein de son entreprise que la politique ne sera pas nécessaire à l'avenir, puis elle la supprime.

Utilisation des informations avant de supprimer des ressources IAM

Vous pouvez utiliser les dernières informations consultées avant de supprimer une ressource IAM, afin de vous assurer qu'un certain délai s'est écoulé depuis qu'une personne a utilisé la ressource pour la dernière fois. Cela s'applique aux utilisateurs, groupes, rôles et politiques. Pour de plus amples informations sur ces actions, veuillez consulter les rubriques suivantes :

- Utilisateurs IAM : Suppression ou désactivation d'un utilisateur IAM
- Groupes : Suppression d'un groupe IAM
- Rôles : Suppression de rôles ou de profils d'instance
- Politiques : Suppression des politiques IAM (détache également la politique des identités)

Utilisation des informations avant de modifier des politiques IAM

Vous pouvez passer en revue les dernières informations consultées d'une identité IAM (utilisateur, groupe ou rôle), ou d'une politique IAM avant de modifier une politique qui affecte la ressource. Ceci est important, car vous ne souhaitez pas supprimer l'accès pour une personne qui l'utilise.

Par exemple, Arnav Desai est développeur et AWS administrateur pour Example Corp. Lorsque son équipe a commencé à utiliser AWS, elle a accordé à tous les développeurs un accès utilisateur avancé qui leur a permis d'accéder pleinement à tous les services sauf IAM et. AWS OrganizationsÀ titre de première étape vers l'attribution du moindre privilège, Arnav souhaite utiliser l'interface AWS CLI pour passer en revue les politiques gérées de son compte.

Pour ce faire, Arnav recense d'abord dans son compte les politiques d'autorisation gérées par le client et qui sont attachées à une identité, à l'aide de la commande suivante :

```
aws iam list-policies --scope Local --only-attached --policy-usage-filter
 PermissionsPolicy
```

À partir de la réponse, il enregistre l'ARN de chaque politique. Arnav génère ensuite un rapport sur les dernières informations consultées pour chaque politique à l'aide de la commande suivante.

```
aws iam generate-service-last-accessed-details --arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Depuis cette réponse, il capture l'ID du rapport généré depuis le champ `JobId`. Arnav interroge ensuite la commande suivante jusqu'à ce que le champ `JobStatus` renvoie la valeur `COMPLETED` ou `FAILED`. Si la tâche a échoué, il capture l'erreur.

```
aws iam get-service-last-accessed-details --job-id 98a765b4-3cde-2101-2345-example678f9
```

Lorsque le statut de la tâche est de`COMPLETED`, Arnav analyse le contenu du JSON-formatted `ServicesLastAccessed` tableau.

```
 "ServicesLastAccessed": [
       {
           "TotalAuthenticatedEntities": 1,
           "LastAuthenticated": 2018-11-01T21:24:33.222Z,
           "ServiceNamespace": "dynamodb",
           "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/IAMExampleUser",
           "ServiceName": "Amazon DynamoDB"
       },

       {
           "TotalAuthenticatedEntities": 0,
           "ServiceNamespace": "ec2",
           "ServiceName": "Amazon EC2"
       },

       {
           "TotalAuthenticatedEntities": 3,
           "LastAuthenticated": 2018-08-25T15:29:51.156Z,
           "ServiceNamespace": "s3",
           "LastAuthenticatedEntity": "arn:aws:iam::123456789012:role/IAMExampleRole",
           "ServiceName": "Amazon S3"
       }
   ]
```

À partir de ces informations, Arnav apprend que la politique `ExamplePolicy1` autorise l'accès à trois services : Amazon DynamoDB, Amazon S3 et Amazon EC2. L'utilisateur IAM nommé `IAMExampleUser` a récemment tenté d'accéder à DynamoDB le 1er novembre, et une personne a utilisé le rôle `IAMExampleRole` pour tenter d'accéder à Amazon S3 le 25 août. Il existe également

deux autres entités qui ont tenté d'accéder à Amazon S3 au cours de cette année. Cependant, personne n'a tenté d'accéder à Amazon EC2 au cours de l'année écoulée.

Cela signifie qu'Arnav peut supprimer en toute sécurité les actions Amazon EC2 de la politique. Arnav souhaite vérifier le document JSON de la politique. Tout d'abord, il doit déterminer le numéro de version de la politique à l'aide de la commande suivante.

```
aws iam list-policy-versions --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Dans la réponse, Arnav recueille le numéro de version actuelle par défaut depuis le tableau `Versions`. Il utilise ensuite le numéro de version (`v2`) pour demander le document de politique JSON à l'aide de la commande suivante.

```
aws iam get-policy-version --policy-arn arn:aws:iam::123456789012:policy/ExamplePolicy1
 --version-id v2
```

Arnav stocke le document de politique JSON renvoyé dans le champ `Document` du tableau `PolicyVersion`. Dans le document de politique, Arnav recherche les actions dans l'espace de noms `ec2`. S'il ne reste pas d'actions d'autres espaces de noms dans la politique, il détache ensuite la politique des identités affectées (utilisateurs, groupes et rôles). Il supprime ensuite la politique. Dans ce cas, la politique inclut les services Amazon DynamoDB et Amazon S3. Par conséquent, Arnav supprime les actions Amazon EC2 du document et enregistre ses modifications. Il utilise ensuite la commande suivante pour mettre à jour la politique à l'aide de la nouvelle version du document et pour définir cette version en tant que version de politique par défaut.

```
aws iam create-policy-version --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1 --policy-document file://UpdatedPolicy.json --set-as-default
```

La politique `ExamplePolicy1` est maintenant mise à jour pour supprimer l'accès au service Amazon EC2 superflu.

Autres scénarios IAM

Les informations sur la date à laquelle une ressource IAM (utilisateur, groupe, rôle ou politique) a tenté pour la dernière fois d'accéder à un service peut vous aider lorsque vous exécutez l'une des tâches suivantes :

- Policies (Politiques) : modification d'une politique en ligne ou gérée par un client pour supprimer les autorisations
- Policies (Politiques) : conversion d'une politique en ligne en une politique gérée, puis suppression
- Policies (Politiques) : ajout d'un refus explicite à une politique existante
- Policies (Politiques) : détachement d'une politique gérée d'une identité (utilisateur, groupe ou rôle)
- Entities (Entités) : définissez une limite d'autorisations pour contrôler les autorisations maximales qu'une entité (utilisateur ou rôle) peut avoir
- Groups (Groupes) : suppression d'utilisateurs d'un groupe

Utilisation des informations pour affiner les autorisations d'une unité d'organisation

Vous pouvez utiliser les dernières informations consultées pour affiner les autorisations d'une unité organisationnelle (UO) dans AWS Organizations.

Par exemple, John Stiles est AWS Organizations administrateur. Il est chargé de veiller à ce que les membres de l'entreprise Comptes AWS ne disposent pas d'autorisations excessives. Dans le cadre d'un contrôle de sécurité périodique, il passe en revue les autorisations de son organisation. Son unité organisationnelle `Development` contient des comptes qui sont souvent utilisés pour tester les nouveaux services AWS . John décide d'examiner périodiquement le rapport concernant les services qui n'ont pas été consultés dans plus de 180 jours. Ensuite, il supprime des autorisations d'accès à ces services pour les membres de l'unité organisationnelle.

John se connecte à la console IAM avec les informations d'identification de son compte de gestion. Dans la console IAM, il localise les AWS Organizations données de l'unité d'organisation. `Development` Il examine le tableau du rapport sur l'accès aux AWS services et constate que deux services n'ont pas été consultés depuis plus de 180 jours qu'il avait préféré. Il se souvient avoir ajouté des autorisations permettant aux équipes de développement d'accéder à Amazon Lex et à Amazon AWS Database Migration Service. John contacte les équipes de développement et vérifie qu'elles n'ont plus besoin de tester ces services.

John est maintenant prêt à agir sur les dernières informations consultées. Il choisit Edit in (Faire des modifications dans) AWS Organizations et le système lui rappelle que la stratégie de contrôle de service est attachée à plusieurs entités. Il choisit Continue (Continuer). Dans AWS Organizations, il passe en revue les cibles pour savoir à quelles AWS Organizations entités le SCP est rattaché. Toutes les entités se trouvent dans l'unité organisationnelle `Development`.

John décide de refuser l'accès à Amazon Lex et les AWS Database Migration Service actions dans le `NewServiceTest` SCP. Cette action supprime l'accès inutile aux services.

Services et actions concernant les dernières informations consultées relatives à une action IAM

Le tableau suivant répertorie les AWS services pour lesquels les informations du dernier accès à l'action IAM sont affichées. Pour obtenir la liste des actions de chaque service, consultez la section Actions, ressources et clés de condition pour les AWS services dans la référence d'autorisation de service.

AWS fournit les dernières informations consultées au format JSON afin de rationaliser l'automatisation des flux de travail de gestion des politiques. Grâce aux informations de référence du service, vous pouvez accéder aux dernières informations sur les actions consultées à Services AWS partir de fichiers lisibles par machine. Pour plus d'informations, consultez les Informations sur les Service AWS simplifiés pour l'accès programmatique dans la Référence des autorisations de service.

Service	Préfixe de service
Gestion des identités et des accès AWS et Access Analyzer	access-analyzer
Gestion de compte AWS	compte
AWS Certificate Manager	acm
Amazon Managed Workflows for Apache Airflow	airflow
AWS Amplify	amplify
AWS Amplify Générateur d'interface utilisateur	amplifyuibuilder
Amazon AppIntegrations	app-integrations
AWS AppConfig	appconfig
Amazon AppFlow	appflow
AWS Profileur des coûts d'application	application-cost-profiler

Service	Préfixe de service
Informations sur les CloudWatch applications Amazon	applicationinsights
AWS App Mesh	appmesh
WorkSpaces Applications Amazon	appstream
AWS AppSync	appsync
Amazon Managed Service for Prometheus	aps
Amazon Athena	athéna
AWS Audit Manager	auditmanager
AWS Auto Scaling	autoscaling
AWS Marketplace	aws-marketplace
AWS Backup	sauvegarde
AWS Batch	lot
Amazon Braket	braket
AWS Budgets	Budgets
AWS Cloud9	Cloud9
AWS CloudFormation	cloudformation
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm
Amazon CloudSearch	cloudsearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch

Service	Préfixe de service
AWS CodeArtifact	codeartifact
AWS CodeDeploy	codedeploy
Amazon CodeGuru Profiler	codeguru-profiler
CodeGuru Réviseur Amazon	codeguru-reviewer
AWS CodePipeline	codepipeline
AWS CodeStar	codestar
AWS CodeStar Notifications	codestar-notifications
Amazon Cognito Identity	cognito-identity
Groupes d'utilisateurs Amazon Cognito	cognito-idp
Amazon Cognito Sync	cognito-sync
Amazon Comprehend Medical	comprehendmedical
Optimiseur de calcul AWS	compute-optimizer
AWS Config	config
Amazon Connect Customer	connexion
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	dataexchange
AWS Data Pipeline	datapipeline
DynamoDB Accelerator	dax

Service	Préfixe de service
AWS Device Farm	devicefarm
Amazon DevOps Guru	devops-guru
AWS Direct Connect	directconnect
Amazon Data Lifecycle Manager	dlm
AWS Database Migration Service	dms
Clusters élastiques Amazon DocumentDB	docdb-elastic
Amazon DynamoDB	dynamodb
Amazon Elastic Block Store	ebs
Amazon Elastic Compute Cloud	ec2
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Public	ecr-public
Amazon Elastic Container Service	ecs
Amazon Elastic Kubernetes Service	eks
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	elasticbeanstalk
Amazon Elastic File System	elasticfilesystem
Elastic Load Balancing	elasticloadbalancing
Amazon EMR sur EKS (conteneurs EMR)	emr-containers
Amazon EMR sans serveur	emr-serverless
Amazon OpenSearch Service	es

Service	Préfixe de service
Amazon EventBridge	events
Amazon FinSpace	finspace
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms
Amazon Fraud Detector	frauddetector
Amazon FSx	fsx
GameLift Serveurs Amazon	gamelift
Amazon Location Service	geo
Amazon Glacier	glacier
Amazon Managed Grafana	grafana
AWS IoT Greengrass	greengrass
AWS Ground Station	groundstation
Amazon GuardDuty	guardduty
AWS HealthLake	healthlake
Amazon Honeycode	honeycode
Gestion des identités et des accès AWS	iam
AWS Boutique d'identités	identitystore
EC2 Image Builder	imagebuilder
Amazon Inspector Classic	inspector

Service	Préfixe de service
Amazon Inspector	inspector2
AWS IoT	iot
AWS IoT Analytics	iotanalytics
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents
AWS IoT Fleet Hub	iotfleethub
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	iottwinmaker
AWS IoT Wireless	iotwireless
Amazon Interactive Video Service	ivs
Service de vidéo interactive Amazon Chat	ivschat
Streaming géré par Amazon pour Apache Kafka	kafka
Amazon Managed Streaming for Kafka Connect	kafkaconnect
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalytics
AWS Key Management Service	kms
AWS Lambda	lambda
Amazon Lex V2	lex
AWS License Manager Gestionnaire des abonnements Linux	license-manager-linux-subscriptions

Service	Préfixe de service
Amazon Lightsail	lightsail
Amazon CloudWatch Logs	journaux
Amazon Lookout for Equipment	lookoutequipment
Amazon Lookout for Metrics	lookoutmetrics
AWS Mainframe Modernization	m2
Amazon Managed Blockchain	managedbl ockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB	base de données de mémoire
AWS Transform MGN	mgn
AWS Migration Hub	mgh
AWS Migration Hub Strategy Recommendations	migration hub-strategy
Amazon Pinpoint	mobiletargeting
Amazon MQ	mq
AWS Network Manager	networkmanager
Amazon Nimble Studio	nimble
AWS HealthOmics	omics

Service	Préfixe de service
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	outposts
AWS Organizations	organisations
AWS Performance Insights	pi
Amazon EventBridge Pipes	pipes
Amazon Polly	polly
Profils des clients Amazon Connect	profile
Amazon QLDB	qldb
AWS Resource Access Manager	ram
AWS Corbeille	rbin
Amazon Relational Database Service	rds
Amazon Redshift	redshift
API de données Amazon Redshift	redshift-data
AWS Migration Hub Refactor Spaces	refactor-spaces
Amazon Rekognition	rekognition
AWS Resilience Hub	resiliencehub
Explorateur de ressources AWS	resource-explorer-2
Groupes de ressources AWS	resource-groups
AWS RoboMaker	robomaker

Service	Préfixe de service
Gestion des identités et des accès AWS Des rôles n'importe où	rolesanywhere
Amazon Route 53	route53
Amazon Route 53 Recovery Controls	route53-recovery-control-config
Amazon Route 53 Recovery Readiness	route53-recovery-readiness
Amazon Route 53 Resolver	route53resolver
AWS CloudWatchRHUM	rum
Amazon Simple Storage Service	s3
Amazon S3 sur Outposts	s3-outposts
Fonctionnalités géospatiales d'Amazon SageMaker AI	sagemaker-geospatial
Savings Plans	savingsplans
EventBridgeSchémas Amazon	schemas
Amazon SimpleDB	sdb
AWS Secrets Manager	secretsmanager
AWS Security Hub CSPM	securityhub
Amazon Security Lake	securitylake
AWS Serverless Application Repository	serverlessrepo
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas

Service	Préfixe de service
Amazon Simple Email Service	ses
AWS Shield	shield
AWS Signer	signer
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
Service de messages SMS et vocaux Amazon Pinpoint	sms-voice
AWS Snowball Edge	snowball
Amazon Simple Queue Service	sqs
AWS Systems Manager	ssm
AWS Systems Manager Incident Manager	ssm-incidents
Gestionnaire de systèmes AWS pour SAP	ssm-sap
AWS Step Functions	states
AWS Security Token Service	sts
Amazon Simple Workflow Service	swf
Amazon CloudWatch Synthetics	synthetics
AWS Resource Groups Tagging API	balise
Amazon Textract	textract
Amazon Timestream	timestream
AWS Générateur de réseaux de télécommunications	tnb
Amazon Transcribe	transcribe

Service	Préfixe de service
AWS Transfer Family	transfert
Amazon Translate	translate
Connect Customer Voice ID	voiceid
Amazon VPC Lattice	vpc-lattice
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Le savoir des clients d'Amazon Connect	wisdom
Amazon WorkLink	worklink
Amazon WorkSpaces	espaces de travail
AWS X-Ray	xray

Actions concernant les dernières informations consultées relatives à une action

Le tableau suivant répertorie les actions pour lesquelles les dernières informations consultées relatives à une action sont disponibles.

Important

L'action `iam:UpdateAccountName` deviendra obsolète le 22 avril 2026. Après le 22 avril 2026, seule l'autorisation `account:PutAccountName` contrôlera l'accès à la mise à jour du nom du compte. Nous vous recommandons vivement de mettre à jour toutes les politiques de contrôle des services (SCP) qui contrôlent les mises à jour des noms de compte pour utiliser l'autorisation `account:PutAccountName`.

Préfixe de service	Actions
access-analyzer	analyseur d'accès : ApplyArchiveRule

Préfixe de service	Actions
	analyseur d'accès : CancelPolicyGeneration
	analyseur d'accès : CheckAccessNotGranted
	analyseur d'accès : CheckNoNewAccess
	analyseur d'accès : CheckNoPublicAccess
	analyseur d'accès : CreateAccessPreview
	analyseur d'accès : CreateAnalyzer
	analyseur d'accès : CreateArchiveRule
	analyseur d'accès : DeleteAnalyzer
	analyseur d'accès : DeleteArchiveRule
	analyseur d'accès : GenerateFindingRecommendation
	analyseur d'accès : GetAccessPreview
	analyseur d'accès : GetAnalyzedResource
	analyseur d'accès : GetAnalyzer
	analyseur d'accès : GetArchiveRule
	analyseur d'accès : GetFinding
	analyseur d'accès : GetFindingRecommendation
	analyseur d'accès : GetFindingsStatistics
	analyseur d'accès : GetGeneratedPolicy
	analyseur d'accès : ListAccessPreviewFindings
	analyseur d'accès : ListAccessPreviews
	analyseur d'accès : ListAnalyzedResources

Préfixe de service	Actions
	analyseur d'accès : ListAnalyzers
	analyseur d'accès : ListArchiveRules
	analyseur d'accès : ListFindings
	analyseur d'accès : ListPolicyGenerations
	analyseur d'accès : StartPolicyGeneration
	analyseur d'accès : StartResourceScan
	analyseur d'accès : UpdateAnalyzer
	analyseur d'accès : UpdateArchiveRule
	analyseur d'accès : UpdateFindings
	analyseur d'accès : ValidatePolicy

Préfixe de service	Actions
compte	compte : AcceptPrimaryEmailUpdate compte : DeleteAlternateContact compte : DisableRegion compte : EnableRegion compte : GetAccountInformation compte : GetAlternateContact compte : GetContactInformation compte : GetGovCloudAccountInformation compte : GetPrimaryEmail compte : GetRegionOptStatus compte : ListRegions compte : PutAccountName compte : PutAlternateContact compte : PutContactInformation compte : StartPrimaryEmailUpdate

Préfixe de service	Actions
acm	ACM : CreateAcmeEndpoint
	ACM : DeleteCertificate
	ACM : DescribeCertificate
	ACM : ExportCertificate
	ACM : GetAccountConfiguration
	ACM : GetCertificate
	ACM : ImportCertificate
	ACM : ListAcmeEndpoints
	ACM : ListCertificates
	ACM : PutAccountConfiguration
	ACM : RenewCertificate
	ACM : RequestCertificate
	ACM : ResendValidationEmail
	ACM : SearchCertificates
	ACM : UpdateCertificateOptions

Préfixe de service	Actions
airflow	débit d'air : CreateCliToken débit d'air : CreateEnvironment débit d'air : CreateWebLoginToken débit d'air : DeleteEnvironment débit d'air : GetEnvironment débit d'air : ListEnvironments débit d'air : PublishMetrics débit d'air : UpdateEnvironment

Préfixe de service	Actions
amplify	amplifier : CreateApp
	amplifier : CreateBackendEnvironment
	amplifier : CreateBranch
	amplifier : CreateDeployment
	amplifier : CreateDomainAssociation
	amplifier : CreateWebHook
	amplifier : DeleteApp
	amplifier : DeleteBackendEnvironment
	amplifier : DeleteBranch
	amplifier : DeleteDomainAssociation
	amplifier : DeleteJob
	amplifier : DeleteWebHook
	amplifier : GenerateAccessLogs
	amplifier : GetApp
	amplifier : GetArtifactUrl
	amplifier : GetBackendEnvironment
	amplifier : GetBranch
	amplifier : GetDomainAssociation
	amplifier : GetJob
	amplifier : GetWebHook
	amplifier : ListApps

Préfixe de service	Actions
	amplifier : ListArtifacts
	amplifier : ListBackendEnvironments
	amplifier : ListBranches
	amplifier : ListDomainAssociations
	amplifier : ListJobs
	amplifier : ListWebHooks
	amplifier : StartDeployment
	amplifier : StartJob
	amplifier : StopJob
	amplifier : UpdateApp
	amplifier : UpdateBranch
	amplifier : UpdateDomainAssociation
	amplifier : UpdateWebHook

Préfixe de service	Actions
amplifyuibuilder	Amplify UI Builder : CreateComponent Amplify UI Builder : CreateForm Amplify UI Builder : CreateTheme Amplify UI Builder : DeleteComponent Amplify UI Builder : DeleteForm Amplify UI Builder : DeleteTheme Amplify UI Builder : ExportComponents Amplify UI Builder : ExportThemes Amplify UI Builder : GetCodegenJob Amplify UI Builder : ListCodegenJobs Amplify UI Builder : ListComponents Amplify UI Builder : ListForms Amplify UI Builder : ListThemes Amplify UI Builder : ResetMetadataFlag Amplify UI Builder : StartCodegenJob Amplify UI Builder : UpdateComponent Amplify UI Builder : UpdateForm Amplify UI Builder : UpdateTheme

Préfixe de service	Actions
app-integrations	intégrations d'applications : CreateApplication
	intégrations d'applications : CreateDataIntegration
	intégrations d'applications : CreateDataIntegrationAssociation
	intégrations d'applications : CreateEventIntegration
	intégrations d'applications : DeleteApplication
	intégrations d'applications : DeleteDataIntegration
	intégrations d'applications : DeleteEventIntegration
	intégrations d'applications : GetApplication
	intégrations d'applications : GetDataIntegration
	intégrations d'applications : GetEventIntegration
	intégrations d'applications : ListApplicationAssociations
	intégrations d'applications : ListApplications
	intégrations d'applications : ListDataIntegrationAssociations
	intégrations d'applications : ListDataIntegrations
	intégrations d'applications : ListEventIntegrationAssociations
	intégrations d'applications : ListEventIntegrations
	intégrations d'applications : UpdateApplication
	intégrations d'applications : UpdateDataIntegration
	intégrations d'applications : UpdateDataIntegrationAssociation
	intégrations d'applications : UpdateEventIntegration

Préfixe de service	Actions
appconfig	configuration de l'application : CreateApplication configuration de l'application : CreateConfigurationProfile configuration de l'application : CreateDeploymentStrategy configuration de l'application : CreateEnvironment configuration de l'application : CreateExtension configuration de l'application : CreateExtensionAssociation configuration de l'application : CreateHostedConfigurationVersion configuration de l'application : DeleteApplication configuration de l'application : DeleteConfigurationProfile configuration de l'application : DeleteDeploymentStrategy configuration de l'application : DeleteEnvironment configuration de l'application : DeleteExtension configuration de l'application : DeleteExtensionAssociation configuration de l'application : DeleteHostedConfigurationVersion configuration de l'application : GetAccountSettings configuration de l'application : GetApplication configuration de l'application : GetConfiguration configuration de l'application : GetConfigurationProfile configuration de l'application : GetDeployment configuration de l'application : GetDeploymentStrategy configuration de l'application : GetEnvironment

Préfixe de service	Actions
	configuration de l'application : GetExtension configuration de l'application : GetExtensionAssociation configuration de l'application : GetHostedConfigurationVersion configuration de l'application : ListApplications configuration de l'application : ListConfigurationProfiles configuration de l'application : ListDeploymentStrategies configuration de l'application : ListDeployments configuration de l'application : ListEnvironments configuration de l'application : ListExtensionAssociations configuration de l'application : ListExtensions configuration de l'application : ListHostedConfigurationVersions configuration de l'application : StartDeployment configuration de l'application : StopDeployment configuration de l'application : UpdateAccountSettings configuration de l'application : UpdateApplication configuration de l'application : UpdateConfigurationProfile configuration de l'application : UpdateDeploymentStrategy configuration de l'application : UpdateEnvironment configuration de l'application : UpdateExtension configuration de l'application : UpdateExtensionAssociation configuration de l'application : ValidateConfiguration

Préfixe de service	Actions
appflow	flux d'applications : CancelFlowExecutions
	flux d'applications : CreateConnectorProfile
	flux d'applications : CreateFlow
	flux d'applications : DeleteConnectorProfile
	flux d'applications : DeleteFlow
	flux d'applications : DescribeConnector
	flux d'applications : DescribeConnectorEntity
	flux d'applications : DescribeConnectorProfiles
	flux d'applications : DescribeConnectors
	flux d'applications : DescribeFlow
	flux d'applications : DescribeFlowExecutionRecords
	flux d'applications : ListConnectorEntities
	flux d'applications : ListConnectors
	flux d'applications : ListFlows
	flux d'applications : RegisterConnector
	flux d'applications : ResetConnectorMetadataCache
	flux d'applications : StartFlow
	flux d'applications : StopFlow
	flux d'applications : UnRegisterConnector
	flux d'applications : UpdateConnectorProfile
	flux d'applications : UpdateConnectorRegistration

Préfixe de service	Actions
	flux d'applications : UpdateFlow

Préfixe de service	Actions
applicationinsights	informations sur les applications : AddWorkload informations sur les applications : CreateApplication informations sur les applications : CreateComponent informations sur les applications : CreateLogPattern informations sur les applications : DeleteApplication informations sur les applications : DeleteComponent informations sur les applications : DeleteLogPattern informations sur les applications : DescribeApplication informations sur les applications : DescribeComponent informations sur les applications : DescribeComponentConfiguration informations sur les applications : DescribeComponentConfigurationRecommendation informations sur les applications : DescribeLogPattern informations sur les applications : DescribeObservation informations sur les applications : DescribeProblem informations sur les applications : DescribeProblemObservations informations sur les applications : DescribeWorkload informations sur les applications : ListApplications informations sur les applications : ListComponents informations sur les applications : ListConfigurationHistory informations sur les applications : ListLogPatternSets

Préfixe de service	Actions
	informations sur les applications : ListLogPatterns informations sur les applications : ListProblems informations sur les applications : ListWorkloads informations sur les applications : RemoveWorkload informations sur les applications : UpdateApplication informations sur les applications : UpdateComponent informations sur les applications : UpdateComponentConfiguration informations sur les applications : UpdateLogPattern informations sur les applications : UpdateWorkload

Préfixe de service	Actions
appmesh	maillage d'applications : CreateGatewayRoute
	maillage d'applications : CreateMesh
	maillage d'applications : CreateRoute
	maillage d'applications : CreateVirtualGateway
	maillage d'applications : CreateVirtualNode
	maillage d'applications : CreateVirtualRouter
	maillage d'applications : CreateVirtualService
	maillage d'applications : DeleteGatewayRoute
	maillage d'applications : DeleteMesh
	maillage d'applications : DeleteRoute
	maillage d'applications : DeleteVirtualGateway
	maillage d'applications : DeleteVirtualNode
	maillage d'applications : DeleteVirtualRouter
	maillage d'applications : DeleteVirtualService
	maillage d'applications : DescribeGatewayRoute
	maillage d'applications : DescribeMesh
	maillage d'applications : DescribeRoute
	maillage d'applications : DescribeVirtualGateway
	maillage d'applications : DescribeVirtualNode
	maillage d'applications : DescribeVirtualRouter
	maillage d'applications : DescribeVirtualService

Préfixe de service	Actions
	maillage d'applications : ListGatewayRoutes
	maillage d'applications : ListMeshes
	maillage d'applications : ListRoutes
	maillage d'applications : ListVirtualGateways
	maillage d'applications : ListVirtualNodes
	maillage d'applications : ListVirtualRouters
	maillage d'applications : ListVirtualServices
	maillage d'applications : StreamAggregatedResources
	maillage d'applications : UpdateGatewayRoute
	maillage d'applications : UpdateMesh
	maillage d'applications : UpdateRoute
	maillage d'applications : UpdateVirtualGateway
	maillage d'applications : UpdateVirtualNode
	maillage d'applications : UpdateVirtualRouter
	maillage d'applications : UpdateVirtualService

Préfixe de service	Actions
appstream	appstream : AssociateAppBlockBuilderAppBlock appstream : AssociateApplicationFleet appstream : AssociateApplicationToEntitlement appstream : AssociateFleet appstream : AssociateSoftwareToImageBuilder appstream : BatchAssociateUserStack appstream : BatchDisassociateUserStack appstream : CopyImage appstream : CreateAppBlock appstream : CreateAppBlockBuilder appstream : CreateAppBlockBuilderStreamingURL appstream : CreateApplication appstream : CreateDirectoryConfig appstream : CreateEntitlement appstream : CreateFleet appstream : CreateImageBuilder appstream : CreateImageBuilderStreamingURL appstream : CreateStack appstream : URL CreateStreaming appstream : CreateThemeForStack appstream : CreateUpdatedImage

Préfixe de service	Actions
	appstream : CreateUsageReportSubscription appstream : CreateUser appstream : DeleteAppBlock appstream : DeleteAppBlockBuilder appstream : DeleteApplication appstream : DeleteDirectoryConfig appstream : DeleteEntitlement appstream : DeleteFleet appstream : DeleteImage appstream : DeleteImageBuilder appstream : DeleteImagePermissions appstream : DeleteStack appstream : DeleteThemeForStack appstream : DeleteUsageReportSubscription appstream : DeleteUser appstream : DescribeAppBlockBuilderAppBlockAssociations appstream : DescribeAppBlockBuilders appstream : DescribeAppBlocks appstream : DescribeAppLicenseUsage appstream : DescribeApplicationFleetAssociations appstream : DescribeApplications

Préfixe de service	Actions
	appstream : DescribeDirectoryConfigs appstream : DescribeEntitlements appstream : DescribeFleets appstream : DescribeImageBuilders appstream : DescribeImagePermissions appstream : DescribeImages appstream : DescribeSessions appstream : DescribeStacks appstream : DescribeThemeForStack appstream : DescribeUsageReportSubscriptions appstream : DescribeUserStackAssociations appstream : DescribeUsers appstream : DisableUser appstream : DisassociateAppBlockBuilderAppBlock appstream : DisassociateApplicationFleet appstream : DisassociateApplicationFromEntitlement appstream : DisassociateFleet appstream : DisassociateSoftwareFromImageBuilder appstream : EnableUser appstream : ExpireSession appstream : GetExportImageTask

Préfixe de service	Actions
	appstream : ListAssociatedFleets appstream : ListAssociatedStacks appstream : ListEntitledApplications appstream : ListExportImageTasks appstream : StartAppBlockBuilder appstream : StartFleet appstream : StartImageBuilder appstream : StartSoftwareDeploymentToImageBuilder appstream : StopAppBlockBuilder appstream : StopFleet appstream : StopImageBuilder appstream : UpdateAppBlockBuilder appstream : UpdateApplication appstream : UpdateDirectoryConfig appstream : UpdateEntitlement appstream : UpdateFleet appstream : UpdateImagePermissions appstream : UpdateStack appstream : UpdateThemeForStack

Préfixe de service	Actions
appsync	synchronisation des applications : AssociateApi
	synchronisation des applications : AssociateMergedGraphqlApi
	synchronisation des applications : AssociateSourceGraphqlApi
	synchronisation des applications : CreateApi
	synchronisation des applications : CreateApiCache
	synchronisation des applications : CreateApiKey
	synchronisation des applications : CreateChannelNamespace
	synchronisation des applications : CreateDataSource
	synchronisation des applications : CreateDomainName
	synchronisation des applications : CreateFunction
	synchronisation des applications : CreateGraphqlApi
	synchronisation des applications : CreateResolver
	synchronisation des applications : CreateType
	synchronisation des applications : DeleteApi
	synchronisation des applications : DeleteApiCache
	synchronisation des applications : DeleteApiKey
	synchronisation des applications : DeleteChannelNamespace
	synchronisation des applications : DeleteDataSource
	synchronisation des applications : DeleteDomainName
	synchronisation des applications : DeleteFunction
	synchronisation des applications : DeleteGraphqlApi

Préfixe de service	Actions
	synchronisation des applications : DeleteResolver
	synchronisation des applications : DeleteType
	synchronisation des applications : DisassociateApi
	synchronisation des applications : DisassociateMergedGraphqlApi
	synchronisation des applications : DisassociateSourceGraphqlApi
	synchronisation des applications : EvaluateCode
	synchronisation des applications : EvaluateMappingTemplate
	synchronisation des applications : FlushApiCache
	synchronisation des applications : GetApi
	synchronisation des applications : GetApiAssociation
	synchronisation des applications : GetApiCache
	synchronisation des applications : GetChannelNamespace
	synchronisation des applications : GetDataSource
	synchronisation des applications : GetDataSourceIntrospection
	synchronisation des applications : GetDomainName
	synchronisation des applications : GetFunction
	synchronisation des applications : GetGraphqlApi
	synchronisation des applications : GetGraphqlApiEnvironmentVariables
	synchronisation des applications : GetIntrospectionSchema
	synchronisation des applications : GetResolver

Préfixe de service	Actions
	synchronisation des applications : GetSchemaCreationStatus
	synchronisation des applications : GetSourceApiAssociation
	synchronisation des applications : GetType
	synchronisation des applications : ListApiKeys
	synchronisation des applications : ListApis
	synchronisation des applications : ListChannelNamespaces
	synchronisation des applications : ListDataSources
	synchronisation des applications : ListDomainNames
	synchronisation des applications : ListFunctions
	synchronisation des applications : ListGraphqlApis
	synchronisation des applications : ListResolvers
	synchronisation des applications : ListResolversByFunction
	synchronisation des applications : ListSourceApiAssociations
	synchronisation des applications : ListTypes
	synchronisation des applications : ListTypesByAssociation
	synchronisation des applications : PutGraphqlApiEnvironmentVariables
	synchronisation des applications : StartDataSourceIntrospection
	synchronisation des applications : StartSchemaCreation
	synchronisation des applications : StartSchemaMerge
	synchronisation des applications : UpdateApi

Préfixe de service	Actions
	synchronisation des applications : UpdateApiCache
	synchronisation des applications : UpdateApiKey
	synchronisation des applications : UpdateChannelNamespace
	synchronisation des applications : UpdateDataSource
	synchronisation des applications : UpdateDomainName
	synchronisation des applications : UpdateFunction
	synchronisation des applications : UpdateGraphqlApi
	synchronisation des applications : UpdateResolver
	synchronisation des applications : UpdateSourceApiAssociation
	synchronisation des applications : UpdateType

Préfixe de service	Actions
aps	Cartes : CreateAlertManagerDefinition Cartes : CreateAnomalyDetector Cartes : CreateLoggingConfiguration Cartes : CreateQueryLoggingConfiguration Cartes : CreateRuleGroupsNamespace Cartes : CreateWorkspace Cartes : DeleteAlertManagerDefinition Cartes : DeleteAnomalyDetector Cartes : DeleteLoggingConfiguration Cartes : DeleteQueryLoggingConfiguration Cartes : DeleteResourcePolicy Cartes : DeleteRuleGroupsNamespace Cartes : DeleteScraper Cartes : DeleteScraperLoggingConfiguration Cartes : DeleteWorkspace Cartes : DescribeAlertManagerDefinition Cartes : DescribeAnomalyDetector Cartes : DescribeLoggingConfiguration Cartes : DescribeQueryLoggingConfiguration Cartes : DescribeResourcePolicy Cartes : DescribeRuleGroupsNamespace

Préfixe de service	Actions
	Cartes : DescribeScraper Cartes : DescribeScraperLoggingConfiguration Cartes : DescribeWorkspace Cartes : DescribeWorkspaceConfiguration Cartes : GetDefaultScraperConfiguration Cartes : ListAnomalyDetectors Cartes : ListRuleGroupsNamespaces Cartes : ListScrapers Cartes : ListWorkspaces Cartes : PutAlertManagerDefinition Cartes : PutAnomalyDetector Cartes : PutResourcePolicy Cartes : PutRuleGroupsNamespace Cartes : UpdateLoggingConfiguration Cartes : UpdateQueryLoggingConfiguration Cartes : UpdateScraper Cartes : UpdateScraperLoggingConfiguration Cartes : UpdateWorkspaceAlias Cartes : UpdateWorkspaceConfiguration

Préfixe de service	Actions
athéna	Athéna : BatchGetNamedQuery
	Athéna : BatchGetPreparedStatement
	Athéna : BatchGetQueryExecution
	Athéna : CancelCapacityReservation
	Athéna : CreateCapacityReservation
	Athéna : CreateDataCatalog
	Athéna : CreateNamedQuery
	Athéna : CreateNotebook
	Athéna : CreatePreparedStatement
	Athéna : CreatePresignedNotebookUrl
	Athéna : CreateWorkGroup
	Athéna : DeleteCapacityReservation
	Athéna : DeleteDataCatalog
	Athéna : DeleteNamedQuery
	Athéna : DeleteNotebook
	Athéna : DeletePreparedStatement
	Athéna : DeleteWorkGroup
	Athéna : ExportNotebook
	Athéna : GetCalculationExecution
	Athéna : GetCalculationExecutionCode
	Athéna : GetCalculationExecutionStatus

Préfixe de service	Actions
	Athéna : GetCapacityAssignmentConfiguration
	Athéna : GetCapacityReservation
	Athéna : GetDataCatalog
	Athéna : GetDatabase
	Athéna : GetNamedQuery
	Athéna : GetNotebookMetadata
	Athéna : GetPreparedStatement
	Athéna : GetQueryExecution
	Athéna : GetQueryResults
	Athéna : GetQueryResultsStream
	Athéna : GetQueryRuntimeStatistics
	Athéna : GetResourceDashboard
	Athéna : GetSession
	Athéna : GetSessionEndpoint
	Athéna : GetSessionStatus
	Athéna : GetTableMetadata
	Athéna : GetWorkGroup
	Athéna : ImportNotebook
	Athéna : ListApplicationDPUSizes
	Athéna : ListCalculationExecutions
	Athéna : ListCapacityReservations

Préfixe de service	Actions
	Athéna : ListDataCatalogs
	Athéna : ListDatabases
	Athéna : ListEngineVersions
	Athéna : ListExecutors
	Athéna : ListNamedQueries
	Athéna : ListNotebookMetadata
	Athéna : ListNotebookSessions
	Athéna : ListPreparedStatements
	Athéna : ListQueryExecutions
	Athéna : ListSessions
	Athéna : ListTableMetadata
	Athéna : ListWorkGroups
	Athéna : PutCapacityAssignmentConfiguration
	Athéna : StartCalculationExecution
	Athéna : StartQueryExecution
	Athéna : StartSession
	Athéna : StopCalculationExecution
	Athéna : StopQueryExecution
	Athéna : TerminateSession
	Athéna : UpdateCapacityReservation
	Athéna : UpdateDataCatalog

Préfixe de service	Actions
	Athéna : UpdateNamedQuery
	Athéna : UpdateNotebook
	Athéna : UpdateNotebookMetadata
	Athéna : UpdatePreparedStatement
	Athéna : UpdateWorkGroup

Préfixe de service	Actions
auditmanager	responsable de l'audit : AssociateAssessmentReportEvidenceFolder responsable de l'audit : BatchAssociateAssessmentReportEvidence responsable de l'audit : BatchCreateDelegationByAssessment responsable de l'audit : BatchDeleteDelegationByAssessment responsable de l'audit : BatchDisassociateAssessmentReportEvidence responsable de l'audit : BatchImportEvidenceToAssessmentControl responsable de l'audit : CreateAssessment responsable de l'audit : CreateAssessmentFramework responsable de l'audit : CreateAssessmentReport responsable de l'audit : CreateControl responsable de l'audit : DeleteAssessment responsable de l'audit : DeleteAssessmentFramework responsable de l'audit : DeleteAssessmentFrameworkShare responsable de l'audit : DeleteAssessmentReport responsable de l'audit : DeleteControl responsable de l'audit : DeregisterAccount responsable de l'audit : DeregisterOrganizationAdminAccount responsable de l'audit : DisassociateAssessmentReportEvidenceFolder responsable de l'audit : GetAccountStatus

Préfixe de service	Actions
	responsable de l'audit : GetAssessment responsable de l'audit : GetAssessmentFramework responsable de l'audit : GetAssessmentReportUrl responsable de l'audit : GetChangeLogs responsable de l'audit : GetControl responsable de l'audit : GetDelegations responsable de l'audit : GetEvidence responsable de l'audit : GetEvidenceByEvidenceFolder responsable de l'audit : GetEvidenceFileUploadUrl responsable de l'audit : GetEvidenceFolder responsable de l'audit : GetEvidenceFoldersByAssessment responsable de l'audit : GetEvidenceFoldersByAssessmentControl responsable de l'audit : GetInsights responsable de l'audit : GetInsightsByAssessment responsable de l'audit : GetOrganizationAdminAccount responsable de l'audit : GetServicesInScope responsable de l'audit : GetSettings responsable de l'audit : ListAssessmentControlInsightsByControlDomain responsable de l'audit : ListAssessmentFrameworkShareRequests responsable de l'audit : ListAssessmentFrameworks

Préfixe de service	Actions
	responsable de l'audit : ListAssessmentReports responsable de l'audit : ListAssessments responsable de l'audit : ListControlDomainInsights responsable de l'audit : ListControlDomainInsightsByAssessment responsable de l'audit : ListControlInsightsByControlDomain responsable de l'audit : ListControls responsable de l'audit : ListKeywordsForDataSource responsable de l'audit : ListNotifications responsable de l'audit : RegisterAccount responsable de l'audit : RegisterOrganizationAdminAccount responsable de l'audit : StartAssessmentFrameworkShare responsable de l'audit : UpdateAssessment responsable de l'audit : UpdateAssessmentControl responsable de l'audit : UpdateAssessmentControlSetStatus responsable de l'audit : UpdateAssessmentFramework responsable de l'audit : UpdateAssessmentFrameworkShare responsable de l'audit : UpdateAssessmentStatus responsable de l'audit : UpdateControl responsable de l'audit : UpdateSettings responsable de l'audit : ValidateAssessmentReportIntegrity

Préfixe de service	Actions
autoscaling	mise à l'échelle automatique : AttachInstances
	mise à l'échelle automatique : AttachLoadBalancerTargetGroups
	mise à l'échelle automatique : AttachLoadBalancers
	mise à l'échelle automatique : AttachTrafficSources
	mise à l'échelle automatique : BatchDeleteScheduledAction
	mise à l'échelle automatique : BatchPutScheduledUpdateGroupAction
	mise à l'échelle automatique : CancelInstanceRefresh
	mise à l'échelle automatique : CompleteLifecycleAction
	mise à l'échelle automatique : CreateAutoScalingGroup
	mise à l'échelle automatique : CreateLaunchConfiguration
	mise à l'échelle automatique : DeleteAutoScalingGroup
	mise à l'échelle automatique : DeleteLaunchConfiguration
	mise à l'échelle automatique : DeleteLifecycleHook
	mise à l'échelle automatique : DeleteNotificationConfiguration
	mise à l'échelle automatique : DeletePolicy
	mise à l'échelle automatique : DeleteScheduledAction
	mise à l'échelle automatique : DeleteWarmPool
	mise à l'échelle automatique : DescribeAccountLimits
	mise à l'échelle automatique : DescribeAdjustmentTypes
	mise à l'échelle automatique : DescribeAutoScalingGroups

Préfixe de service	Actions
	mise à l'échelle automatique : DescribeAutoScalingInstances
	mise à l'échelle automatique : DescribeAutoScalingNotificationTypes
	mise à l'échelle automatique : DescribeInstanceRefreshes
	mise à l'échelle automatique : DescribeLaunchConfigurations
	mise à l'échelle automatique : DescribeLifecycleHookTypes
	mise à l'échelle automatique : DescribeLifecycleHooks
	mise à l'échelle automatique : DescribeLoadBalancerTargetGroups
	mise à l'échelle automatique : DescribeLoadBalancers
	mise à l'échelle automatique : DescribeMetricCollectionTypes
	mise à l'échelle automatique : DescribeNotificationConfigurations
	mise à l'échelle automatique : DescribePolicies
	mise à l'échelle automatique : DescribeScalingActivities
	mise à l'échelle automatique : DescribeScalingProcessTypes
	mise à l'échelle automatique : DescribeScheduledActions
	mise à l'échelle automatique : DescribeTerminationPolicyTypes
	mise à l'échelle automatique : DescribeTrafficSources
	mise à l'échelle automatique : DescribeWarmPool
	mise à l'échelle automatique : DetachInstances
	mise à l'échelle automatique : DetachLoadBalancerTargetGroups
	mise à l'échelle automatique : DetachLoadBalancers

Préfixe de service	Actions
	mise à l'échelle automatique : DetachTrafficSources
	mise à l'échelle automatique : DisableMetricsCollection
	mise à l'échelle automatique : EnableMetricsCollection
	mise à l'échelle automatique : EnterStandby
	mise à l'échelle automatique : ExecutePolicy
	mise à l'échelle automatique : ExitStandby
	mise à l'échelle automatique : GetPredictiveScalingForecast
	mise à l'échelle automatique : PutLifecycleHook
	mise à l'échelle automatique : PutNotificationConfiguration
	mise à l'échelle automatique : PutScalingPolicy
	mise à l'échelle automatique : PutScheduledUpdateGroupAction
	mise à l'échelle automatique : PutWarmPool
	mise à l'échelle automatique : RecordLifecycleActionHeartbeat
	mise à l'échelle automatique : ResumeProcesses
	mise à l'échelle automatique : RollbackInstanceRefresh
	mise à l'échelle automatique : SetDesiredCapacity
	mise à l'échelle automatique : SetInstanceHealth
	mise à l'échelle automatique : SetInstanceProtection
	mise à l'échelle automatique : StartInstanceRefresh
	mise à l'échelle automatique : SuspendProcesses

Préfixe de service	Actions
	mise à l'échelle automatique : TerminateInstanceInAutoScalingGroup mise à l'échelle automatique : UpdateAutoScalingGroup
aws-marketplace	AWS Marketplace : GetEntitlements

Préfixe de service	Actions
sauvegarde	sauvegarde : CancelLegalHold
	sauvegarde : CreateBackupPlan
	sauvegarde : CreateBackupSelection
	sauvegarde : CreateBackupVault
	sauvegarde : CreateFramework
	sauvegarde : CreateLegalHold
	sauvegarde : CreateLogicallyAirGappedBackupVault
	sauvegarde : CreateReportPlan
	sauvegarde : CreateRestoreAccessBackupVault
	sauvegarde : CreateRestoreTestingPlan
	sauvegarde : CreateRestoreTestingSelection
	sauvegarde : CreateTieringConfiguration
	sauvegarde : DeleteBackupPlan
	sauvegarde : DeleteBackupSelection
	sauvegarde : DeleteBackupVault
	sauvegarde : DeleteBackupVaultAccessPolicy
	sauvegarde : DeleteBackupVaultLockConfiguration
	sauvegarde : DeleteBackupVaultNotifications
	sauvegarde : DeleteFramework
	sauvegarde : DeleteRecoveryPoint
	sauvegarde : DeleteReportPlan

Préfixe de service	Actions
	sauvegarde : DeleteRestoreTestingPlan
	sauvegarde : DeleteRestoreTestingSelection
	sauvegarde : DeleteTieringConfiguration
	sauvegarde : DescribeBackupJob
	sauvegarde : DescribeBackupVault
	sauvegarde : DescribeCopyJob
	sauvegarde : DescribeFramework
	sauvegarde : DescribeGlobalSettings
	sauvegarde : DescribeProtectedResource
	sauvegarde : DescribeRecoveryPoint
	sauvegarde : DescribeRegionSettings
	sauvegarde : DescribeReportJob
	sauvegarde : DescribeReportPlan
	sauvegarde : DescribeRestoreJob
	sauvegarde : DescribeScanJob
	sauvegarde : DisassociateRecoveryPoint
	sauvegarde : DisassociateRecoveryPointFromParent
	sauvegarde : ExportBackupPlanTemplate
	sauvegarde : GetBackupPlan
	sauvegarde : GetBackupPlanFromJSON
	sauvegarde : GetBackupPlanFromTemplate

Préfixe de service	Actions
	sauvegarde : GetBackupSelection
	sauvegarde : GetBackupVaultAccessPolicy
	sauvegarde : GetBackupVaultNotifications
	sauvegarde : GetLegalHold
	sauvegarde : GetRecoveryPointRestoreMetadata
	sauvegarde : GetRestoreJobMetadata
	sauvegarde : GetRestoreTestingInferredMetadata
	sauvegarde : GetRestoreTestingPlan
	sauvegarde : GetRestoreTestingSelection
	sauvegarde : GetSupportedResourceTypes
	sauvegarde : GetTieringConfiguration
	sauvegarde : ListBackupJobSummaries
	sauvegarde : ListBackupJobs
	sauvegarde : ListBackupPlanTemplates
	sauvegarde : ListBackupPlanVersions
	sauvegarde : ListBackupPlans
	sauvegarde : ListBackupSelections
	sauvegarde : ListBackupVaults
	sauvegarde : ListCopyJobSummaries
	sauvegarde : ListCopyJobs
	sauvegarde : ListFrameworks

Préfixe de service	Actions
	sauvegarde : ListIndexedRecoveryPoints
	sauvegarde : ListLegalHolds
	sauvegarde : ListProtectedResources
	sauvegarde : ListRecoveryPointsByBackupVault
	sauvegarde : ListRecoveryPointsByLegalHold
	sauvegarde : ListRecoveryPointsByResource
	sauvegarde : ListReportJobs
	sauvegarde : ListReportPlans
	sauvegarde : ListRestoreAccessBackupVaults
	sauvegarde : ListRestoreJobSummaries
	sauvegarde : ListRestoreJobs
	sauvegarde : ListRestoreJobsByProtectedResource
	sauvegarde : ListRestoreTestingPlans
	sauvegarde : ListRestoreTestingSelections
	sauvegarde : ListScanJobSummaries
	sauvegarde : ListScanJobs
	sauvegarde : ListTieringConfigurations
	sauvegarde : PutBackupVaultAccessPolicy
	sauvegarde : PutBackupVaultLockConfiguration
	sauvegarde : PutBackupVaultNotifications
	sauvegarde : PutRestoreValidationResult

Préfixe de service	Actions
	sauvegarde : StartBackupJob
	sauvegarde : StartCopyJob
	sauvegarde : StartReportJob
	sauvegarde : StartRestoreJob
	sauvegarde : StopBackupJob
	sauvegarde : UpdateBackupPlan
	sauvegarde : UpdateFramework
	sauvegarde : UpdateGlobalSettings
	sauvegarde : UpdateRecoveryPointLifecycle
	sauvegarde : UpdateRegionSettings
	sauvegarde : UpdateReportPlan
	sauvegarde : UpdateRestoreTestingPlan
	sauvegarde : UpdateRestoreTestingSelection
	sauvegarde : UpdateTieringConfiguration

Préfixe de service	Actions
lot	lot : CancelJob lot : CreateComputeEnvironment lot : CreateConsumableResource lot : CreateJobQueue lot : CreateQuotaShare lot : CreateSchedulingPolicy lot : CreateServiceEnvironment lot : DeleteComputeEnvironment lot : DeleteConsumableResource lot : DeleteJobQueue lot : DeleteQuotaShare lot : DeleteSchedulingPolicy lot : DeleteServiceEnvironment lot : DeregisterJobDefinition lot : DescribeComputeEnvironments lot : DescribeConsumableResource lot : DescribeJobDefinitions lot : DescribeJobQueues lot : DescribeJobs lot : DescribeQuotaShare lot : DescribeSchedulingPolicies

Préfixe de service	Actions
	lot : DescribeServiceEnvironments lot : DescribeServiceJob lot : GetJobQueueSnapshot lot : ListConsumableResources lot : ListJobs lot : ListJobsByConsumableResource lot : ListQuotaShares lot : ListSchedulingPolicies lot : ListServiceJobs lot : RegisterJobDefinition lot : SubmitJob lot : SubmitServiceJob lot : TerminateJob lot : TerminateServiceJob lot : UpdateComputeEnvironment lot : UpdateConsumableResource lot : UpdateJobQueue lot : UpdateQuotaShare lot : UpdateSchedulingPolicy lot : UpdateServiceEnvironment lot : UpdateServiceJob

Préfixe de service	Actions
braket	support : CancelJob support : CancelQuantumTask support : CreateJob support : CreateQuantumTask support : CreateSpendingLimit support : GetDevice support : GetJob support : GetQuantumTask support : SearchDevices support : SearchJobs support : SearchQuantumTasks support : SearchSpendingLimits

Préfixe de service	Actions
Budgets	budgets : CreateBudgetAction budgets : DeleteBudgetAction budgets : DescribeBudgetAction budgets : DescribeBudgetActionHistories budgets : DescribeBudgetActionsForAccount budgets : DescribeBudgetActionsForBudget budgets : ExecuteBudgetAction budgets : ModifyBudget budgets : UpdateBudgetAction budgets : ViewBudget
Cloud9	cloud 9 : EC2 CreateEnvironment nuage 9 : CreateEnvironmentMembership nuage 9 : DeleteEnvironment nuage 9 : DeleteEnvironmentMembership nuage 9 : DescribeEnvironmentMemberships nuage 9 : DescribeEnvironmentStatus nuage 9 : DescribeEnvironments nuage 9 : ListEnvironments nuage 9 : UpdateEnvironment nuage 9 : UpdateEnvironmentMembership

Préfixe de service	Actions
cloudformation	formation des nuages : BatchDescribeTypeConfigurations formation des nuages : CancelUpdateStack formation des nuages : ContinueUpdateRollback formation des nuages : CreateChangeSet formation des nuages : CreateGeneratedTemplate formation des nuages : CreateStack formation des nuages : CreateStackInstances formation des nuages : CreateStackSet formation des nuages : DeactivateType formation des nuages : DeleteChangeSet formation des nuages : DeleteGeneratedTemplate formation des nuages : DeleteStack formation des nuages : DeleteStackInstances formation des nuages : DeleteStackSet formation des nuages : DeregisterType formation des nuages : DescribeAccountLimits formation des nuages : DescribeChangeSet formation des nuages : DescribeChangeSetHooks formation des nuages : DescribeEvents formation des nuages : DescribeGeneratedTemplate formation des nuages : DescribeOrganizationsAccess

Préfixe de service	Actions
	formation des nuages : DescribePublisher
	formation des nuages : DescribeResourceScan
	formation des nuages : DescribeStackDriftDetectionStatus
	formation des nuages : DescribeStackEvents
	formation des nuages : DescribeStackInstance
	formation des nuages : DescribeStackResource
	formation des nuages : DescribeStackResourceDrifts
	formation des nuages : DescribeStackResources
	formation des nuages : DescribeStackSet
	formation des nuages : DescribeStackSetOperation
	formation des nuages : DescribeStacks
	formation des nuages : DescribeType
	formation des nuages : DescribeTypeRegistration
	formation des nuages : DetectStackDrift
	formation des nuages : DetectStackResourceDrift
	formation des nuages : DetectStackSetDrift
	formation des nuages : EstimateTemplateCost
	formation des nuages : ExecuteChangeSet
	formation des nuages : GetGeneratedTemplate
	formation des nuages : GetHookResult
	formation des nuages : GetStackPolicy

Préfixe de service	Actions
	formation des nuages : GetTemplate
	formation des nuages : GetTemplateSummary
	formation des nuages : ImportStacksToStackSet
	formation des nuages : ListChangeSets
	formation des nuages : ListExports
	formation des nuages : ListGeneratedTemplates
	formation des nuages : ListHookResults
	formation des nuages : ListImports
	formation des nuages : ListResourceScanRelatedResources
	formation des nuages : ListResourceScanResources
	formation des nuages : ListResourceScans
	formation des nuages : ListStackInstanceResourceDrifts
	formation des nuages : ListStackInstances
	formation des nuages : ListStackRefactors
	formation des nuages : ListStackResources
	formation des nuages : ListStackSetAutoDeploymentTargets
	formation des nuages : ListStackSetOperationResults
	formation des nuages : ListStackSetOperations
	formation des nuages : ListStackSets
	formation des nuages : ListTypeRegistrations
	formation des nuages : ListTypeVersions

Préfixe de service	Actions
	formation des nuages : ListTypes formation des nuages : PublishType formation des nuages : RecordHandlerProgress formation des nuages : RegisterPublisher formation des nuages : RegisterType formation des nuages : RollbackStack formation des nuages : SetStackPolicy formation des nuages : SetTypeConfiguration formation des nuages : SetTypeDefaultVersion formation des nuages : SignalResource formation des nuages : StartResourceScan formation des nuages : StopStackSetOperation formation des nuages : TestType formation des nuages : UpdateGeneratedTemplate formation des nuages : UpdateStack formation des nuages : UpdateStackInstances formation des nuages : UpdateStackSet formation des nuages : UpdateTerminationProtection formation des nuages : ValidateTemplate

Préfixe de service	Actions
cloudfront	front de nuage : AssociateAlias
	front de nuage : AssociateDistributionTenantWebACL
	front de nuage : AssociateDistributionWebACL
	front de nuage : CreateCachePolicy
	front de nuage : CreateCloudFrontOriginAccessIdentity
	front de nuage : CreateConnectionFunction
	front de nuage : CreateContinuousDeploymentPolicy
	front de nuage : CreateDistributionTenant
	front de nuage : CreateFieldLevelEncryptionConfig
	front de nuage : CreateFieldLevelEncryptionProfile
	front de nuage : CreateFunction
	front de nuage : CreateInvalidation
	front de nuage : CreateKeyGroup
	front de nuage : CreateKeyValueStore
	front de nuage : CreateMonitoringSubscription
	front de nuage : CreateOriginAccessControl
	front de nuage : CreateOriginRequestPolicy
	front de nuage : CreatePublicKey
	front de nuage : CreateRealtimeLogConfig
	front de nuage : CreateResponseHeadersPolicy
	front de nuage : CreateTrustStore

Préfixe de service	Actions
	front de nuage : DeleteAnycastIpList
	front de nuage : DeleteCachePolicy
	front de nuage : DeleteCloudFrontOriginAccessIdentity
	front de nuage : DeleteConnectionFunction
	front de nuage : DeleteConnectionGroup
	front de nuage : DeleteContinuousDeploymentPolicy
	front de nuage : DeleteDistribution
	front de nuage : DeleteDistributionTenant
	front de nuage : DeleteFieldLevelEncryptionConfig
	front de nuage : DeleteFieldLevelEncryptionProfile
	front de nuage : DeleteFunction
	front de nuage : DeleteKeyGroup
	front de nuage : DeleteKeyValueStore
	front de nuage : DeleteMonitoringSubscription
	front de nuage : DeleteOriginAccessControl
	front de nuage : DeleteOriginRequestPolicy
	front de nuage : DeletePublicKey
	front de nuage : DeleteRealtimeLogConfig
	front de nuage : DeleteResponseHeadersPolicy
	front de nuage : DeleteStreamingDistribution
	front de nuage : DeleteTrustStore

Préfixe de service	Actions
	front de nuage : DeleteVpcOrigin
	front de nuage : DescribeFunction
	front de nuage : DescribeKeyValueStore
	front de nuage : DisassociateDistributionTenantWebACL
	front de nuage : DisassociateDistributionWebACL
	front de nuage : GetAnycastIpList
	front de nuage : GetCachePolicy
	front de nuage : GetCachePolicyConfig
	front de nuage : GetCloudFrontOriginAccessIdentity
	front de nuage : GetCloudFrontOriginAccessIdentityConfig
	front de nuage : GetContinuousDeploymentPolicy
	front de nuage : GetContinuousDeploymentPolicyConfig
	front de nuage : GetDistributionConfig
	front de nuage : GetFieldLevelEncryption
	front de nuage : GetFieldLevelEncryptionConfig
	front de nuage : GetFieldLevelEncryptionProfile
	front de nuage : GetFieldLevelEncryptionProfileConfig
	front de nuage : GetFunction
	front de nuage : GetInvalidation
	front de nuage : GetInvalidationForDistributionTenant
	front de nuage : GetKeyGroup

Préfixe de service	Actions
	front de nuage : GetKeyGroupConfig
	front de nuage : GetMonitoringSubscription
	front de nuage : GetOriginAccessControl
	front de nuage : GetOriginAccessControlConfig
	front de nuage : GetOriginRequestPolicy
	front de nuage : GetOriginRequestPolicyConfig
	front de nuage : GetPublicKey
	front de nuage : GetPublicKeyConfig
	front de nuage : GetRealtimeLogConfig
	front de nuage : GetResponseHeadersPolicy
	front de nuage : GetResponseHeadersPolicyConfig
	front de nuage : GetStreamingDistribution
	front de nuage : GetStreamingDistributionConfig
	front de nuage : GetVpcOrigin
	front de nuage : ListAnycastIpLists
	front de nuage : ListCachePolicies
	front de nuage : ListCloudFrontOriginAccessIdentities
	front de nuage : ListConflictingAliases
	front de nuage : ListConnectionFunctions
	front de nuage : ListConnectionGroups
	front de nuage : ListContinuousDeploymentPolicies

Préfixe de service	Actions
	front de nuage : ListDistributionTenants
	front de nuage : ListDistributionTenantsByCustomization
	front de nuage : ListDistributions
	front de nuage : ListDistributionsByAnycastIpListId
	front de nuage : ListDistributionsByCachePolicyId
	front de nuage : ListDistributionsByConnectionMode
	front de nuage : ListDistributionsByKeyGroup
	front de nuage : ListDistributionsByOriginRequestPolicyId
	front de nuage : ListDistributionsByRealtimeLogConfig
	front de nuage : ListDistributionsByResponseHeadersPolicyId
	front de nuage : ListDistributionsByVpcOriginId
	front de nuage : ListDistributionsByWebACLId
	front de nuage : ListFieldLevelEncryptionConfigs
	front de nuage : ListFieldLevelEncryptionProfiles
	front de nuage : ListFunctions
	front de nuage : ListInvalidations
	front de nuage : ListInvalidationsForDistributionTenant
	front de nuage : ListKeyGroups
	front de nuage : ListKeyValueStores
	front de nuage : ListOriginAccessControls
	front de nuage : ListOriginRequestPolicies

Préfixe de service	Actions
	front de nuage : ListPublicKeys
	front de nuage : ListRealtimeLogConfigs
	front de nuage : ListResponseHeadersPolicies
	front de nuage : ListStreamingDistributions
	front de nuage : ListTrustStores
	front de nuage : PublishConnectionFunction
	front de nuage : PublishFunction
	front de nuage : TestConnectionFunction
	front de nuage : TestFunction
	front de nuage : UpdateAnycastIpList
	front de nuage : UpdateCachePolicy
	front de nuage : UpdateCloudFrontOriginAccessIdentity
	front de nuage : UpdateConnectionFunction
	front de nuage : UpdateConnectionGroup
	front de nuage : UpdateContinuousDeploymentPolicy
	front de nuage : UpdateDistribution
	front de nuage : UpdateDistributionTenant
	front de nuage : UpdateFieldLevelEncryptionConfig
	front de nuage : UpdateFieldLevelEncryptionProfile
	front de nuage : UpdateFunction
	front de nuage : UpdateKeyGroup

Préfixe de service	Actions
	front de nuage : UpdateKeyValueStore front de nuage : UpdateOriginAccessControl front de nuage : UpdateOriginRequestPolicy front de nuage : UpdatePublicKey front de nuage : UpdateRealtimeLogConfig front de nuage : UpdateResponseHeadersPolicy front de nuage : UpdateTrustStore
cloudhsm	cloudhsm : CreateHsm cloudhsm : DeleteBackup cloudhsm : DeleteHsm cloudhsm : DeleteResourcePolicy cloudhsm : DescribeBackups cloudhsm : DescribeClusters cloudhsm : GetResourcePolicy cloudhsm : InitializeCluster cloudhsm : ModifyBackupAttributes cloudhsm : ModifyCluster cloudhsm : PutResourcePolicy cloudhsm : RestoreBackup

Préfixe de service	Actions
cloudsearch	recherche dans le cloud : BuildSuggesters
	recherche dans le cloud : CreateDomain
	recherche dans le cloud : DefineAnalysisScheme
	recherche dans le cloud : DefineExpression
	recherche dans le cloud : DefineIndexField
	recherche dans le cloud : DefineSuggester
	recherche dans le cloud : DeleteAnalysisScheme
	recherche dans le cloud : DeleteDomain
	recherche dans le cloud : DeleteExpression
	recherche dans le cloud : DeleteIndexField
	recherche dans le cloud : DeleteSuggester
	recherche dans le cloud : DescribeAnalysisSchemes
	recherche dans le cloud : DescribeAvailabilityOptions
	recherche dans le cloud : DescribeDomainEndpointOptions
	recherche dans le cloud : DescribeDomains
	recherche dans le cloud : DescribeExpressions
	recherche dans le cloud : DescribeIndexFields
	recherche dans le cloud : DescribeScalingParameters
	recherche dans le cloud : DescribeServiceAccessPolicies
	recherche dans le cloud : DescribeSuggesters
	recherche dans le cloud : IndexDocuments

Préfixe de service	Actions
	recherche dans le cloud : ListDomainNames
	recherche dans le cloud : UpdateAvailabilityOptions
	recherche dans le cloud : UpdateDomainEndpointOptions
	recherche dans le cloud : UpdateScalingParameters
	recherche dans le cloud : UpdateServiceAccessPolicies

Préfixe de service	Actions
cloudtrail	traînée nuageuse : CancelQuery traînée nuageuse : CreateChannel traînée nuageuse : CreateDashboard traînée nuageuse : CreateEventDataStore traînée nuageuse : CreateTrail traînée nuageuse : DeleteChannel traînée nuageuse : DeleteDashboard traînée nuageuse : DeleteEventDataStore traînée nuageuse : DeleteResourcePolicy traînée nuageuse : DeleteTrail traînée nuageuse : DeregisterOrganizationDelegatedAdmin traînée nuageuse : DescribeQuery traînée nuageuse : DescribeTrails traînée nuageuse : DisableFederation traînée nuageuse : GenerateQuery traînée nuageuse : GetChannel traînée nuageuse : GetDashboard traînée nuageuse : GetEventConfiguration traînée nuageuse : GetEventDataStore traînée nuageuse : GetEventDataStoreData traînée nuageuse : GetEventSelectors

Préfixe de service	Actions
	traînée nuageuse : GetImport traînée nuageuse : GetInsightSelectors traînée nuageuse : GetResourcePolicy traînée nuageuse : GetTrail traînée nuageuse : GetTrailStatus traînée nuageuse : ListChannels traînée nuageuse : ListDashboards traînée nuageuse : ListEventDataStores traînée nuageuse : ListImportFailures traînée nuageuse : ListImports traînée nuageuse : ListInsightsData traînée nuageuse : ListPublicKeys traînée nuageuse : ListQueries traînée nuageuse : ListTrails traînée nuageuse : LookupEvents traînée nuageuse : PutEventConfiguration traînée nuageuse : PutEventSelectors traînée nuageuse : PutInsightSelectors traînée nuageuse : PutResourcePolicy traînée nuageuse : RegisterOrganizationDelegatedAdmin traînée nuageuse : RestoreEventDataStore

Préfixe de service	Actions
	traînée nuageuse : SearchSampleQueries traînée nuageuse : StartEventDataStoreIngestion traînée nuageuse : StartImport traînée nuageuse : StartLogging traînée nuageuse : StartQuery traînée nuageuse : StopEventDataStoreIngestion traînée nuageuse : StopImport traînée nuageuse : StopLogging traînée nuageuse : UpdateChannel traînée nuageuse : UpdateDashboard traînée nuageuse : UpdateEventDataStore traînée nuageuse : UpdateTrail

Préfixe de service	Actions
cloudwatch	surveillance des nuages : DeleteAlarmMuteRule
	surveillance des nuages : DeleteAlarms
	surveillance des nuages : DeleteAnomalyDetector
	surveillance des nuages : DeleteDashboards
	surveillance des nuages : DeleteInsightRules
	surveillance des nuages : DeleteMetricStream
	surveillance des nuages : DescribeAlarmHistory
	surveillance des nuages : DescribeAlarms
	surveillance des nuages : DescribeAlarmsForMetric
	surveillance des nuages : DescribeAnomalyDetectors
	surveillance des nuages : DescribeInsightRules
	surveillance des nuages : DisableAlarmActions
	surveillance des nuages : DisableInsightRules
	surveillance des nuages : EnableAlarmActions
	surveillance des nuages : EnableInsightRules
	surveillance des nuages : GetAlarmMuteRule
	surveillance des nuages : GetDashboard
	surveillance des nuages : GetDataset
	surveillance des nuages : GetInsightRuleReport
	surveillance des nuages : GetMetricStatistics
	surveillance des nuages : GetMetricStream

Préfixe de service	Actions
	surveillance des nuages : ListAlarmMuteRules
	surveillance des nuages : ListDashboards
	surveillance des nuages : ListManagedInsightRules
	surveillance des nuages : ListMetricStreams
	surveillance des nuages : PutAlarmMuteRule
	surveillance des nuages : PutAnomalyDetector
	surveillance des nuages : PutCompositeAlarm
	surveillance des nuages : PutDashboard
	surveillance des nuages : PutInsightRule
	surveillance des nuages : PutLogAlarm
	surveillance des nuages : PutManagedInsightRules
	surveillance des nuages : PutMetricAlarm
	surveillance des nuages : PutMetricStream
	surveillance des nuages : SetAlarmState
	surveillance des nuages : StartMetricStreams
	surveillance des nuages : StopMetricStreams

Préfixe de service	Actions
codeartifact	artefact de code : AssociateExternalConnection artefact de code : CopyPackageVersions artefact de code : CreateDomain artefact de code : CreateRepository artefact de code : DeleteDomain artefact de code : DeleteDomainPermissionsPolicy artefact de code : DeletePackage artefact de code : DeletePackageVersions artefact de code : DeleteRepository artefact de code : DeleteRepositoryPermissionsPolicy artefact de code : DescribeDomain artefact de code : DescribePackage artefact de code : DescribePackageVersion artefact de code : DescribeRepository artefact de code : DisassociateExternalConnection artefact de code : DisposePackageVersions artefact de code : GetAssociatedPackageGroup artefact de code : GetAuthorizationToken artefact de code : GetDomainPermissionsPolicy artefact de code : GetPackageVersionAsset artefact de code : GetPackageVersionReadme

Préfixe de service	Actions
	artefact de code : GetRepositoryEndpoint artefact de code : GetRepositoryPermissionsPolicy artefact de code : ListDomains artefact de code : ListPackageGroups artefact de code : ListPackageVersionAssets artefact de code : ListPackageVersionDependencies artefact de code : ListPackageVersions artefact de code : ListPackages artefact de code : ListRepositories artefact de code : ListRepositoriesInDomain artefact de code : PublishPackageVersion artefact de code : PutDomainPermissionsPolicy artefact de code : PutPackageMetadata artefact de code : PutPackageOriginConfiguration artefact de code : PutRepositoryPermissionsPolicy artefact de code : ReadFromRepository artefact de code : UpdatePackageVersionsStatus artefact de code : UpdateRepository

Préfixe de service	Actions
codedeploy	déploiement de code : BatchGetApplicationRevisions
	déploiement de code : BatchGetApplications
	déploiement de code : BatchGetDeploymentGroups
	déploiement de code : BatchGetDeploymentInstances
	déploiement de code : BatchGetDeploymentTargets
	déploiement de code : BatchGetDeployments
	déploiement de code : BatchGetOnPremisesInstances
	déploiement de code : ContinueDeployment
	déploiement de code : CreateApplication
	déploiement de code : CreateDeployment
	déploiement de code : CreateDeploymentConfig
	déploiement de code : CreateDeploymentGroup
	déploiement de code : DeleteApplication
	déploiement de code : DeleteDeploymentConfig
	déploiement de code : DeleteDeploymentGroup
	déploiement de code : DeleteGitHubAccountToken
	déploiement de code : DeleteResourcesByExternalId
	déploiement de code : DeregisterOnPremisesInstance
	déploiement de code : GetApplication
	déploiement de code : GetApplicationRevision
	déploiement de code : GetDeployment

Préfixe de service	Actions
	déploiement de code : GetDeploymentConfig
	déploiement de code : GetDeploymentGroup
	déploiement de code : GetDeploymentInstance
	déploiement de code : GetDeploymentTarget
	déploiement de code : GetOnPremisesInstance
	déploiement de code : ListApplicationRevisions
	déploiement de code : ListApplications
	déploiement de code : ListDeploymentConfigs
	déploiement de code : ListDeploymentGroups
	déploiement de code : ListDeploymentInstances
	déploiement de code : ListDeploymentTargets
	déploiement de code : ListDeployments
	déploiement de code : ListGitHubAccountTokenNames
	déploiement de code : ListOnPremisesInstances
	déploiement de code : PutLifecycleEventHookExecutionStatus
	déploiement de code : RegisterApplicationRevision
	déploiement de code : RegisterOnPremisesInstance
	déploiement de code : SkipWaitTimeForInstanceTermination
	déploiement de code : StopDeployment
	déploiement de code : UpdateApplication
	déploiement de code : UpdateDeploymentGroup

Préfixe de service	Actions
codeguru-profiler	profileur de code : AddNotificationChannels profileur de code : BatchGetFrameMetricData profileur de code : CreateProfilingGroup profileur de code : DeleteProfilingGroup profileur de code : DescribeProfilingGroup profileur de code : GetFindingsReportAccountSummary profileur de code : GetNotificationConfiguration profileur de code : GetPolicy profileur de code : GetProfile profileur de code : GetRecommendations profileur de code : ListFindingsReports profileur de code : ListProfileTimes profileur de code : ListProfilingGroups profileur de code : PutPermission profileur de code : RemoveNotificationChannel profileur de code : RemovePermission profileur de code : SubmitFeedback profileur de code : UpdateProfilingGroup

Préfixe de service	Actions
codeguru-reviewer	réviseur de code : AssociateRepository
	réviseur de code : CreateCodeReview
	réviseur de code : DescribeCodeReview
	réviseur de code : DescribeRecommendationFeedback
	réviseur de code : DescribeRepositoryAssociation
	réviseur de code : DisassociateRepository
	réviseur de code : ListCodeReviews
	réviseur de code : ListRecommendationFeedback
	réviseur de code : ListRecommendations
	réviseur de code : ListRepositoryAssociations
	réviseur de code : PutRecommendationFeedback

Préfixe de service	Actions
codepipeline	pipeline de code : AcknowledgeJob pipeline de code : AcknowledgeThirdPartyJob pipeline de code : CreateCustomActionType pipeline de code : CreatePipeline pipeline de code : DeleteCustomActionType pipeline de code : DeletePipeline pipeline de code : DeleteWebhook pipeline de code : DeregisterWebhookWithThirdParty pipeline de code : GetActionType pipeline de code : GetJobDetails pipeline de code : GetPipeline pipeline de code : GetPipelineExecution pipeline de code : GetPipelineState pipeline de code : GetThirdPartyJobDetails pipeline de code : ListActionExecutions pipeline de code : ListActionTypes pipeline de code : ListPipelineExecutions pipeline de code : ListPipelines pipeline de code : ListRuleExecutions pipeline de code : ListRuleTypes pipeline de code : ListWebhooks

Préfixe de service	Actions
	pipeline de code : OverrideStageCondition pipeline de code : PollForJobs pipeline de code : PollForThirdPartyJobs pipeline de code : PutActionRevision pipeline de code : PutApprovalResult pipeline de code : PutJobFailureResult pipeline de code : PutJobSuccessResult pipeline de code : PutThirdPartyJobFailureResult pipeline de code : PutThirdPartyJobSuccessResult pipeline de code : PutWebhook pipeline de code : RegisterWebhookWithThirdParty pipeline de code : RollbackStage pipeline de code : StartPipelineExecution pipeline de code : StopPipelineExecution pipeline de code : UpdateActionType pipeline de code : UpdatePipeline

Préfixe de service	Actions
codestar	codestar : AssociateTeamMember codestar : CreateProject codestar : CreateUserProfile codestar : DeleteProject codestar : DeleteUserProfile codestar : DescribeProject codestar : DescribeUserProfile codestar : DisassociateTeamMember codestar : ListProjects codestar : ListResources codestar : ListTeamMembers codestar : ListUserProfiles codestar : UpdateProject codestar : UpdateTeamMember codestar : UpdateUserProfile

Préfixe de service	Actions
codestar-notifications	notifications codestar : CreateNotificationRule notifications codestar : DeleteNotificationRule notifications codestar : DeleteTarget notifications codestar : DescribeNotificationRule notifications codestar : ListEventTypes notifications codestar : ListNotificationRules notifications codestar : ListTargets codestar-notifications:Subscribe codestar-notifications:Unsubscribe notifications codestar : UpdateNotificationRule

Préfixe de service	Actions
cognito-identity	identité cognitive : CreateIdentityPool
	identité cognitive : DeleteIdentities
	identité cognitive : DeleteIdentityPool
	identité cognitive : DescribeIdentity
	identité cognitive : DescribeIdentityPool
	identité cognitive : GetIdentityPoolRoles
	identité cognitive : ListIdentities
	identité cognitive : ListIdentityPools
	identité cognitive : LookupDeveloperIdentity
	identité cognitive : MergeDeveloperIdentities
	identité cognitive : SetIdentityPoolRoles
	identité cognitive : UnlinkDeveloperIdentity
	identité cognitive : UpdateIdentityPool

Préfixe de service	Actions
cognito-idp	identifiant : AddCustomAttributes identifiant : AddUserPoolClientSecret identifiant : AdminAddUserToGroup identifiant : AdminConfirmSignUp identifiant : AdminCreateUser identifiant : AdminDeleteUser identifiant : AdminDeleteUserAttributes identifiant : AdminDisableProviderForUser identifiant : AdminDisableUser identifiant : AdminEnableUser identifiant : AdminForgetDevice identifiant : AdminGetDevice identifiant : AdminGetUser identifiant : AdminGetUserAuthFactors identifiant : AdminInitiateAuth identifiant : AdminLinkProviderForUser identifiant : AdminListDevices identifiant : AdminListGroupsForUser identifiant : AdminListUserAuthEvents identifiant : AdminRemoveUserFromGroup identifiant : AdminResetUserPassword

Préfixe de service	Actions
	identifiant : AdminRespondToAuthChallenge identifiant : AdminSetUserMFAPreference identifiant : AdminSetUserPassword identifiant : AdminSetUserSettings identifiant : AdminUpdateAuthEventFeedback identifiant : AdminUpdateDeviceStatus identifiant : AdminUpdateUserAttributes identifiant : AdminUserGlobalSignOut identifiant : AssociateSoftwareToken identifiant : ChangePassword identifiant : ConfirmDevice identifiant : ConfirmForgotPassword identifiant : ConfirmSignUp identifiant : CreateGroup identifiant : CreateIdentityProvider identifiant : CreateManagedLoginBranding identifiant : CreateResourceServer identifiant : CreateTerms identifiant : CreateUserImportJob identifiant : CreateUserPool identifiant : CreateUserPoolClient

Préfixe de service	Actions
	identifiant : CreateUserPoolDomain identifiant : CreateUserPoolReplica identifiant : DeleteGroup identifiant : DeleteIdentityProvider identifiant : DeleteManagedLoginBranding identifiant : DeleteResourceServer identifiant : DeleteTerms identifiant : DeleteUser identifiant : DeleteUserAttributes identifiant : DeleteUserPool identifiant : DeleteUserPoolClient identifiant : DeleteUserPoolClientSecret identifiant : DeleteUserPoolDomain identifiant : DeleteUserPoolReplica identifiant : DescribeIdentityProvider identifiant : DescribeManagedLoginBranding identifiant : DescribeManagedLoginBrandingByClient identifiant : DescribeResourceServer identifiant : DescribeRiskConfiguration identifiant : DescribeTerms identifiant : DescribeUserImportJob

Préfixe de service	Actions
	identifiant : DescribeUserPool identifiant : DescribeUserPoolClient identifiant : DescribeUserPoolDomain identifiant : ForgetDevice identifiant : ForgotPassword cognito-idp:GetCSVHeader identifiant : GetDevice identifiant : GetGroup identifiant : GetIdentityProviderByIdentifier identifiant : GetLogDeliveryConfiguration identifiant : GetProvisionedLimit identifiant : GetSigningCertificate cognito-idp:GetUICustomization identifiant : GetUser identifiant : GetUserAttributeVerificationCode identifiant : GetUserPoolMfaConfig identifiant : GlobalSignOut identifiant : InitiateAuth identifiant : ListDevices identifiant : ListGroups identifiant : ListIdentityProviders

Préfixe de service	Actions
	identifiant : ListResourceServers identifiant : ListTerms identifiant : ListUserImportJobs identifiant : ListUserPoolClientSecrets identifiant : ListUserPoolClients identifiant : ListUserPoolReplicas identifiant : ListUserPools identifiant : ListUsers identifiant : ListUsersInGroup identifiant : ResendConfirmationCode identifiant : RespondToAuthChallenge identifiant : RevokeToken identifiant : SetLogDeliveryConfiguration identifiant : SetRiskConfiguration cognito-idp:SetUICustomization identifiant : SetUserMFAPreference identifiant : SetUserPoolMfaConfig identifiant : SetUserSettings identifiant : SignUp identifiant : StartUserImportJob identifiant : StopUserImportJob

Préfixe de service	Actions
	identifiant : UpdateAuthEventFeedback identifiant : UpdateDeviceStatus identifiant : UpdateGroup identifiant : UpdateIdentityProvider identifiant : UpdateProvisionedLimit identifiant : UpdateResourceServer identifiant : UpdateTerms identifiant : UpdateUserAttributes identifiant : UpdateUserPool identifiant : UpdateUserPoolClient identifiant : UpdateUserPoolDomain identifiant : UpdateUserPoolReplica identifiant : VerifySoftwareToken identifiant : VerifyUserAttribute

Préfixe de service	Actions
cognito-sync	synchronisation cognitive : BulkPublish
	synchronisation cognitive : DeleteDataset
	synchronisation cognitive : DescribeDataset
	synchronisation cognitive : DescribeIdentityPoolUsage
	synchronisation cognitive : DescribeIdentityUsage
	synchronisation cognitive : GetBulkPublishDetails
	synchronisation cognitive : GetCognitoEvents
	synchronisation cognitive : GetIdentityPoolConfiguration
	synchronisation cognitive : ListDatasets
	synchronisation cognitive : ListIdentityPoolUsage
	synchronisation cognitive : ListRecords
	synchronisation cognitive : RegisterDevice
	synchronisation cognitive : SetCognitoEvents
	synchronisation cognitive : SetIdentityPoolConfiguration
	synchronisation cognitive : SubscribeToDataset
	synchronisation cognitive : UnsubscribeFromDataset
	synchronisation cognitive : UpdateRecords

Préfixe de service	Actions
comprehendmedical	médecine complète : DescribeEntitiesDetectionV2Job médecine complète : DescribeICD10CMInferenceJob médecine complète : DescribePHIDetectionJob médecine complète : DescribeRxNormInferenceJob médecine complète : DescribeSNOMEDCTInferenceJob medical complet : V2 DetectEntities comprehendmedical:DetectPHI comprehendmedical:InferICD10CM médecine complète : InferRxNorm comprehendmedical:InferSNOMEDCT médecine complète : ListEntitiesDetectionV2Jobs médecine complète : ListICD10CMInferenceJobs médecine complète : ListPHIDetectionJobs médecine complète : ListRxNormInferenceJobs médecine complète : ListSNOMEDCTInferenceJobs médecine complète : StartEntitiesDetectionV2Job médecine complète : StartICD10CMInferenceJob médecine complète : StartPHIDetectionJob médecine complète : StartRxNormInferenceJob médecine complète : StartSNOMEDCTInferenceJob médecine complète : StopEntitiesDetectionV2Job

Préfixe de service	Actions
	médecine complète : StopICD10CMInferenceJob médecine complète : StopPHIDetectionJob médecine complète : StopRxNormInferenceJob médecine complète : StopSNOMEDCTInferenceJob

Préfixe de service	Actions
compute-optimizer	optimiseur de calcul : DeleteRecommendationPreferences
	optimiseur de calcul : DescribeRecommendationExportJobs
	optimiseur de calcul : ExportAutoScalingGroupRecommendations
	optimiseur de calcul : ExportEBSVolumeRecommendations
	optimiseur de calcul : ExportEC2InstanceRecommendations
	optimiseur de calcul : ExportECSServiceRecommendations
	optimiseur de calcul : ExportIdleRecommendations
	optimiseur de calcul : ExportLambdaFunctionRecommendations
	optimiseur de calcul : ExportLicenseRecommendations
	optimiseur de calcul : ExportRDSDatabaseRecommendations
	optimiseur de calcul : GetEC2RecommendationProjectedMetrics
	optimiseur de calcul : GetECSServiceRecommendationProjectedMetrics
	optimiseur de calcul : GetEffectiveRecommendationPreferences
	optimiseur de calcul : GetEnrollmentStatus
	optimiseur de calcul : GetEnrollmentStatusesForOrganization
	optimiseur de calcul : GetRDSDatabaseRecommendationProjectedMetrics
	optimiseur de calcul : GetRecommendationPreferences
	optimiseur de calcul : GetRecommendationSummaries
	optimiseur de calcul : PutRecommendationPreferences

Préfixe de service	Actions
	optimiseur de calcul : UpdateEnrollmentStatus

Préfixe de service	Actions
config	configuration : BatchGetResourceConfig configuration : DeleteAggregationAuthorization configuration : DeleteConfigRule configuration : DeleteConfigurationAggregator configuration : DeleteConfigurationRecorder configuration : DeleteConformancePack configuration : DeleteDeliveryChannel configuration : DeleteEvaluationResults configuration : DeleteOrganizationConfigRule configuration : DeleteOrganizationConformancePack configuration : DeletePendingAggregationRequest configuration : DeleteRemediationConfiguration configuration : DeleteRemediationExceptions configuration : DeleteResourceConfig configuration : DeleteRetentionConfiguration configuration : DeleteStoredQuery configuration : DeliverConfigSnapshot configuration : DescribeAggregateComplianceByConfigRules configuration : DescribeAggregateComplianceByConformancePacks configuration : DescribeAggregationAuthorizations

Préfixe de service	Actions
	configuration : DescribeComplianceByConfigRule configuration : DescribeComplianceByResource configuration : DescribeConfigRuleEvaluationStatus configuration : DescribeConfigRules configuration : DescribeConfigurationAggregatorSourcesStatus configuration : DescribeConfigurationAggregators configuration : DescribeConfigurationRecorderStatus configuration : DescribeConfigurationRecorders configuration : DescribeConformancePackCompliance configuration : DescribeConformancePackStatus configuration : DescribeConformancePacks configuration : DescribeDeliveryChannelStatus configuration : DescribeDeliveryChannels configuration : DescribeOrganizationConfigRuleStatuses configuration : DescribeOrganizationConfigRules configuration : DescribeOrganizationConformancePackStatuses configuration : DescribeOrganizationConformancePacks configuration : DescribePendingAggregationRequests configuration : DescribeRemediationConfigurations configuration : DescribeRemediationExceptions configuration : DescribeRemediationExecutionStatus

Préfixe de service	Actions
	configuration : DescribeRetentionConfigurations
	configuration : GetComplianceDetailsByConfigRule
	configuration : GetComplianceDetailsByResource
	configuration : GetComplianceSummaryByConfigRule
	configuration : GetComplianceSummaryByResourceType
	configuration : GetConformancePackComplianceDetails
	configuration : GetConformancePackComplianceSummary
	configuration : GetCustomRulePolicy
	configuration : GetDiscoveredResourceCounts
	configuration : GetOrganizationConfigRuleDetailedStatus
	configuration : GetOrganizationConformancePackDetailedStatus
	configuration : GetOrganizationCustomRulePolicy
	configuration : GetResourceConfigHistory
	configuration : GetResourceEvaluationSummary
	configuration : GetStoredQuery
	configuration : ListConfigurationRecorders
	configuration : ListConformancePackComplianceScores
	configuration : ListConnectors
	configuration : ListDiscoveredResources
	configuration : ListResourceEvaluations
	configuration : ListStoredQueries

Préfixe de service	Actions
	configuration : PutConfigRule configuration : PutConfigurationAggregator configuration : PutConfigurationRecorder configuration : PutConformancePack configuration : PutConnector configuration : PutDeliveryChannel configuration : PutEvaluations configuration : PutExternalEvaluation configuration : PutOrganizationConfigRule configuration : PutOrganizationConformancePack configuration : PutRemediationConfigurations configuration : PutRemediationExceptions configuration : PutResourceConfig configuration : PutRetentionConfiguration configuration : PutStoredQuery configuration : SelectResourceConfig configuration : StartConfigRulesEvaluation configuration : StartConfigurationRecorder configuration : StartRemediationExecution configuration : StartResourceEvaluation configuration : StopConfigurationRecorder

Préfixe de service	Actions
connexion	connecter : ActivateEvaluationForm
	connecter : AssociateAnalyticsDataSet
	connecter : AssociateApprovedOrigin
	connecter : AssociateBot
	connecter : AssociateContactWithUser
	connecter : AssociateDefaultVocabulary
	connecter : AssociateEmailAddressAlias
	connecter : AssociateFlow
	connecter : AssociateInstanceStorageConfig
	connecter : AssociateLambdaFunction
	connecter : AssociateLexBot
	connecter : AssociatePhoneNumberContactFlow
	connecter : AssociateQueueEmailAddresses
	connecter : AssociateQueueQuickConnects
	connecter : AssociateRoutingProfileQueues
	connecter : AssociateSecurityKey
	connecter : AssociateUserProficiencies
	connecter : BatchAssociateAnalyticsDataSet
	connecter : BatchCreateDataTableValue
	connecter : BatchDeleteDataTableValue
	connecter : BatchDescribeDataTableValue

Préfixe de service	Actions
	connecter : BatchDisassociateAnalyticsDataSet
	connecter : BatchGetFlowAssociation
	connecter : BatchPutContact
	connecter : BatchUpdateDataTableValue
	connecter : ClaimPhoneNumber
	connecter : CreateAgentStatus
	connecter : CreateAttachedFile
	connecter : CreateContact
	connecter : CreateContactFlow
	connecter : CreateContactFlowModule
	connecter : CreateContactFlowModuleAlias
	connecter : CreateContactFlowModuleVersion
	connecter : CreateContactFlowVersion
	connecter : CreateDataTable
	connecter : CreateDataTableAttribute
	connecter : CreateEmailAddress
	connecter : CreateEvaluationForm
	connecter : CreateHoursOfOperation
	connecter : CreateInstance
	connecter : CreateIntegrationAssociation
	connecter : CreateParticipant

Préfixe de service	Actions
	connecter : CreatePersistentContactAssociation
	connecter : CreatePredefinedAttribute
	connecter : CreatePrompt
	connecter : CreatePushNotificationRegistration
	connecter : CreateQueue
	connecter : CreateQuickConnect
	connecter : CreateRoutingProfile
	connecter : CreateRule
	connecter : CreateSecurityProfile
	connecter : CreateTaskTemplate
	connecter : CreateTrafficDistributionGroup
	connecter : CreateUseCase
	connecter : CreateUser
	connecter : CreateUserHierarchyGroup
	connecter : CreateView
	connecter : CreateViewVersion
	connecter : CreateVocabulary
	connecter : CreateWorkspace
	connecter : DeactivateEvaluationForm
	connecter : DeleteContactEvaluation
	connecter : DeleteContactFlow

Préfixe de service	Actions
	connecter : DeleteContactFlowModule
	connecter : DeleteContactFlowModuleAlias
	connecter : DeleteContactFlowModuleVersion
	connecter : DeleteContactFlowVersion
	connecter : DeleteDataTable
	connecter : DeleteDataTableAttribute
	connecter : DeleteEmailAddress
	connecter : DeleteEvaluationForm
	connecter : DeleteHoursOfOperation
	connecter : DeleteHoursOfOperationOverride
	connecter : DeleteInstance
	connecter : DeleteIntegrationAssociation
	connecter : DeleteNotification
	connecter : DeletePredefinedAttribute
	connecter : DeletePrompt
	connecter : DeletePushNotificationRegistration
	connecter : DeleteQueue
	connecter : DeleteQuickConnect
	connecter : DeleteRoutingProfile
	connecter : DeleteRule
	connecter : DeleteSecurityProfile

Préfixe de service	Actions
	connecter : DeleteTaskTemplate
	connecter : DeleteTrafficDistributionGroup
	connecter : DeleteUseCase
	connecter : DeleteUser
	connecter : DeleteUserHierarchyGroup
	connecter : DeleteView
	connecter : DeleteVocabulary
	connecter : DeleteWorkspace
	connecter : DeleteWorkspaceMedia
	connecter : DescribeAttachedFilesConfiguration
	connecter : DescribeAuthenticationProfile
	connecter : DescribeContactFlowModuleAlias
	connecter : DescribeDataTableAttribute
	connecter : DescribeHoursOfOperationOverride
	connecter : DescribeInstanceAttribute
	connecter : DescribeInstanceStorageConfig
	connecter : DescribeNotification
	connecter : DescribePhoneNumber
	connecter : DescribeRule
	connecter : DescribeTrafficDistributionGroup
	connecter : DescribeUserHierarchyStructure

Préfixe de service	Actions
	connecter : DescribeVocabulary
	connecter : DisassociateAnalyticsDataSet
	connecter : DisassociateApprovedOrigin
	connecter : DisassociateBot
	connecter : DisassociateEmailAddressAlias
	connecter : DisassociateFlow
	connecter : DisassociateInstanceStorageConfig
	connecter : DisassociateLambdaFunction
	connecter : DisassociateLexBot
	connecter : DisassociatePhoneNumberContactFlow
	connecter : DisassociateQueueEmailAddresses
	connecter : DisassociateQueueQuickConnects
	connecter : DisassociateRoutingProfileQueues
	connecter : DisassociateSecurityKey
	connecter : DisassociateUserProficiencies
	connecter : DismissUserContact
	connecter : EvaluateDataTableValues
	connecter : GetContactAttributes
	connecter : GetContactMetrics
	connecter : GetCurrentMetricData
	connecter : GetCurrentUserData

Préfixe de service	Actions
	connecter : GetEffectiveHoursOfOperations
	connecter : GetEvaluationFormValidation
	connecter : GetFederationToken
	connecter : GetFlowAssociation
	connecter : GetMetricData
	connecter : GetMetricDataV2
	connecter : GetPromptFile
	connecter : GetTaskTemplate
	connecter : GetTrafficDistribution
	connecter : ImportPhoneNumber
	connecter : ImportWorkspaceMedia
	connecter : ListAnalyticsDataAssociations
	connecter : ListAnalyticsDataLakeDataSets
	connecter : ListApprovedOrigins
	connecter : ListAssociatedContacts
	connecter : ListAttachedFilesConfigurations
	connecter : ListAuthenticationProfiles
	connecter : ListBots
	connecter : ListContactEvaluations
	connecter : ListContactFlowModuleAliases
	connecter : ListContactFlowModuleVersions

Préfixe de service	Actions
	connecter : ListContactFlowModules
	connecter : ListContactFlowVersions
	connecter : ListContactFlows
	connecter : ListContactReferences
	connecter : ListDataTableAttributes
	connecter : ListDataTablePrimaryValues
	connecter : ListDataTableValues
	connecter : ListDataTables
	connecter : ListDefaultVocabularies
	connecter : ListEvaluationFormVersions
	connecter : ListEvaluationForms
	connecter : ListFlowAssociations
	connecter : ListHoursOfOperations
	connecter : ListInstanceAttributes
	connecter : ListInstanceStorageConfigs
	connecter : ListIntegrationAssociations
	connecter : ListLambdaFunctions
	connecter : ListLexBots
	connecter : ListNotifications
	connecter : ListPhoneNumbers
	connecter : ListPhoneNumbersV2

Préfixe de service	Actions
	connecter : ListPredefinedAttributes
	connecter : ListPrompts
	connecter : ListQueueEmailAddresses
	connecter : ListQueueQuickConnects
	connecter : ListQueues
	connecter : ListQuickConnects
	connecter : ListRealtimeContactAnalysisSegmentsV2
	connecter : ListRoutingProfileManualAssignmentQueues
	connecter : ListRoutingProfileQueues
	connecter : ListRoutingProfiles
	connecter : ListRules
	connecter : ListSecurityKeys
	connecter : ListSecurityProfileApplications
	connecter : ListSecurityProfileFlowModules
	connecter : ListSecurityProfilePermissions
	connecter : ListSecurityProfiles
	connecter : ListTaskTemplates
	connecter : ListTrafficDistributionGroups
	connecter : ListUseCases
	connecter : ListUserHierarchyGroups
	connecter : ListUserNotifications

Préfixe de service	Actions
	connecter : ListUsers
	connecter : ListViewVersions
	connecter : ListViews
	connecter : ListWorkspaceMedia
	connecter : ListWorkspacePages
	connecter : ListWorkspaces
	connecter : MonitorContact
	connecter : PauseContact
	connecter : PutUserStatus
	connecter : ReleasePhoneNumber
	connecter : ReplicateInstance
	connecter : ResumeContact
	connecter : ResumeContactRecording
	connecter : SearchAgentStatuses
	connecter : SearchAvailablePhoneNumbers
	connecter : SearchContactEvaluations
	connecter : SearchContactFlowModules
	connecter : SearchContactFlows
	connecter : SearchContacts
	connecter : SearchDataTables
	connecter : SearchEmailAddresses

Préfixe de service	Actions
	connecter : SearchEvaluationForms
	connecter : SearchHoursOfOperations
	connecter : SearchPredefinedAttributes
	connecter : SearchPrompts
	connecter : SearchQueues
	connecter : SearchQuickConnects
	connecter : SearchRoutingProfiles
	connecter : SearchRules
	connecter : SearchSecurityProfiles
	connecter : SearchUserHierarchyGroups
	connecter : SearchViews
	connecter : SearchVocabularies
	connecter : SearchWorkspaceAssociations
	connecter : SearchWorkspaces
	connecter : SendChatIntegrationEvent
	connecter : SendOutboundEmail
	connecter : StartChatContact
	connecter : StartContactConversationalAnalyticsJob
	connecter : StartContactEvaluation
	connecter : StartContactMediaProcessing
	connecter : StartContactRecording

Préfixe de service	Actions
	connecter : StartContactStreaming
	connecter : StartEmailContact
	connecter : StartOutboundChatContact
	connecter : StartOutboundEmailContact
	connecter : StartOutboundVoiceContact
	connecter : StartScreenSharing
	connecter : StartTaskContact
	connecter : StartWebRTCContact
	connecter : StopContact
	connecter : StopContactMediaProcessing
	connecter : StopContactRecording
	connecter : StopContactStreaming
	connecter : SubmitContactEvaluation
	connecter : SuspendContactRecording
	connecter : TransferContact
	connecter : UpdateAgentStatus
	connecter : UpdateAttachedFilesConfiguration
	connecter : UpdateAuthenticationProfile
	connecter : UpdateContact
	connecter : UpdateContactAttributes
	connecter : UpdateContactEvaluation

Préfixe de service	Actions
	connecter : UpdateContactFlowContent
	connecter : UpdateContactFlowMetadata
	connecter : UpdateContactFlowModuleAlias
	connecter : UpdateContactFlowModuleContent
	connecter : UpdateContactFlowModuleMetadata
	connecter : UpdateContactFlowName
	connecter : UpdateContactRoutingData
	connecter : UpdateContactSchedule
	connecter : UpdateDataTableAttribute
	connecter : UpdateDataTableMetadata
	connecter : UpdateDataTablePrimaryValues
	connecter : UpdateEmailAddressMetadata
	connecter : UpdateEvaluationForm
	connecter : UpdateHoursOfOperation
	connecter : UpdateHoursOfOperationOverride
	connecter : UpdateInstanceAttribute
	connecter : UpdateInstanceStorageConfig
	connecter : UpdateNotificationContent
	connecter : UpdateParticipantAuthentication
	connecter : UpdateParticipantRoleConfig
	connecter : UpdatePhoneNumber

Préfixe de service	Actions
	connecter : UpdatePhoneNumberMetadata
	connecter : UpdatePredefinedAttribute
	connecter : UpdatePrompt
	connecter : UpdateQueueHoursOfOperation
	connecter : UpdateQueueMaxContacts
	connecter : UpdateQueueName
	connecter : UpdateQueueOutboundCallerConfig
	connecter : UpdateQueueOutboundEmailConfig
	connecter : UpdateQueueStatus
	connecter : UpdateQuickConnectConfig
	connecter : UpdateQuickConnectName
	connecter : UpdateRoutingProfileAgentAvailabilityTimer
	connecter : UpdateRoutingProfileConcurrency
	connecter : UpdateRoutingProfileDefaultOutboundQueue
	connecter : UpdateRoutingProfileName
	connecter : UpdateRoutingProfileQueues
	connecter : UpdateRule
	connecter : UpdateSecurityProfile
	connecter : UpdateTaskTemplate
	connecter : UpdateTrafficDistribution
	connecter : UpdateUserHierarchy

Préfixe de service	Actions
	connecter : UpdateUserHierarchyGroupName connecter : UpdateUserHierarchyStructure connecter : UpdateUserIdentityInfo connecter : UpdateUserNotificationStatus connecter : UpdateUserPhoneConfig connecter : UpdateUserProficiencies connecter : UpdateUserRoutingProfile connecter : UpdateUserSecurityProfiles connecter : UpdateViewContent connecter : UpdateViewMetadata connecter : UpdateWorkspaceMetadata connecter : UpdateWorkspaceTheme connecter : UpdateWorkspaceVisibility
cur	cœur : DeleteReportDefinition cœur : DescribeReportDefinitions cœur : ModifyReportDefinition cœur : PutReportDefinition

Préfixe de service	Actions
databrew	base de données : BatchDeleteRecipeVersion
	base de données : CreateDataset
	base de données : CreateProfileJob
	base de données : CreateProject
	base de données : CreateRecipe
	base de données : CreateRecipeJob
	base de données : CreateRuleset
	base de données : CreateSchedule
	base de données : DeleteDataset
	base de données : DeleteJob
	base de données : DeleteProject
	base de données : DeleteRecipeVersion
	base de données : DeleteRuleset
	base de données : DeleteSchedule
	base de données : DescribeDataset
	base de données : DescribeJob
	base de données : DescribeJobRun
	base de données : DescribeProject
	base de données : DescribeRecipe
	base de données : DescribeRuleset
	base de données : DescribeSchedule

Préfixe de service	Actions
	base de données : ListDatasets
	base de données : ListJobRuns
	base de données : ListJobs
	base de données : ListProjects
	base de données : ListRecipeVersions
	base de données : ListRecipes
	base de données : ListRulesets
	base de données : ListSchedules
	base de données : PublishRecipe
	base de données : SendProjectSessionAction
	base de données : StartJobRun
	base de données : StartProjectSession
	base de données : StopJobRun
	base de données : UpdateDataset
	base de données : UpdateProfileJob
	base de données : UpdateProject
	base de données : UpdateRecipe
	base de données : UpdateRecipeJob
	base de données : UpdateRuleset
	base de données : UpdateSchedule

Préfixe de service	Actions
dataexchange	échange de données : AcceptDataGrant échange de données : CancelJob échange de données : CreateDataGrant échange de données : CreateDataSet échange de données : CreateEventAction échange de données : CreateJob échange de données : CreateRevision échange de données : DeleteAsset échange de données : DeleteDataGrant échange de données : DeleteEventAction échange de données : DeleteRevision échange de données : GetDataGrant échange de données : GetEventAction échange de données : GetJob échange de données : GetReceivedDataGrant échange de données : ListDataGrants échange de données : ListDataSetRevisions échange de données : ListDataSets échange de données : ListEventActions échange de données : ListJobs échange de données : ListReceivedDataGrants

Préfixe de service	Actions
	échange de données : ListRevisionAssets échange de données : RevokeRevision échange de données : SendDataSetNotification échange de données : StartJob échange de données : UpdateAsset échange de données : UpdateDataSet échange de données : UpdateEventAction échange de données : UpdateRevision

Préfixe de service	Actions
datapipeline	pipeline de données : ActivatePipeline
	pipeline de données : CreatePipeline
	pipeline de données : DeactivatePipeline
	pipeline de données : DeletePipeline
	pipeline de données : DescribeObjects
	pipeline de données : DescribePipelines
	pipeline de données : EvaluateExpression
	pipeline de données : GetPipelineDefinition
	pipeline de données : ListPipelines
	pipeline de données : PollForTask
	pipeline de données : PutPipelineDefinition
	pipeline de données : QueryObjects
	pipeline de données : ReportTaskProgress
	pipeline de données : ReportTaskRunnerHeartbeat
	pipeline de données : SetStatus
	pipeline de données : SetTaskStatus
	pipeline de données : ValidatePipelineDefinition

Préfixe de service	Actions
dax	télécopieur : CreateCluster télécopieur : DecreaseReplicationFactor télécopieur : DeleteCluster télécopieur : DeleteParameterGroup télécopieur : DeleteSubnetGroup télécopieur : DescribeClusters télécopieur : DescribeDefaultParameters télécopieur : DescribeEvents télécopieur : DescribeParameterGroups télécopieur : DescribeParameters télécopieur : DescribeSubnetGroups télécopieur : IncreaseReplicationFactor télécopieur : RebootNode télécopieur : UpdateCluster télécopieur : UpdateParameterGroup télécopieur : UpdateSubnetGroup

Préfixe de service	Actions
devicefarm	ferme d'appareils : CreateDevicePool
	ferme d'appareils : CreateInstanceProfile
	ferme d'appareils : CreateNetworkProfile
	ferme d'appareils : CreateProject
	ferme d'appareils : CreateRemoteAccessSession
	ferme d'appareils : CreateTestGridProject
	ferme d'appareils : CreateTestGridUrl
	ferme d'appareils : CreateUpload
	devicefarm:CreateVPCEConfiguration
	ferme d'appareils : DeleteDevicePool
	ferme d'appareils : DeleteInstanceProfile
	ferme d'appareils : DeleteNetworkProfile
	ferme d'appareils : DeleteProject
	ferme d'appareils : DeleteRemoteAccessSession
	ferme d'appareils : DeleteRun
	ferme d'appareils : DeleteTestGridProject
	ferme d'appareils : DeleteUpload
	devicefarm:DeleteVPCEConfiguration
	ferme d'appareils : GetAccountSettings
	ferme d'appareils : GetDevice
	ferme d'appareils : GetDeviceInstance

Préfixe de service	Actions
	ferme d'appareils : GetDevicePool
	ferme d'appareils : GetDevicePoolCompatibility
	ferme d'appareils : GetInstanceProfile
	ferme d'appareils : GetJob
	ferme d'appareils : GetNetworkProfile
	ferme d'appareils : GetOfferingStatus
	ferme d'appareils : GetProject
	ferme d'appareils : GetRemoteAccessSession
	ferme d'appareils : GetRun
	ferme d'appareils : GetSuite
	ferme d'appareils : GetTest
	ferme d'appareils : GetTestGridProject
	ferme d'appareils : GetTestGridSession
	ferme d'appareils : GetUpload
	devicefarm:GetVPCEConfiguration
	ferme d'appareils : ListArtifacts
	ferme d'appareils : ListDeviceInstances
	ferme d'appareils : ListDevicePools
	ferme d'appareils : ListDevices
	ferme d'appareils : ListInstanceProfiles
	ferme d'appareils : ListJobs

Préfixe de service	Actions
	ferme d'appareils : ListNetworkProfiles
	ferme d'appareils : ListOfferingPromotions
	ferme d'appareils : ListOfferingTransactions
	ferme d'appareils : ListOfferings
	ferme d'appareils : ListProjects
	ferme d'appareils : ListRemoteAccessSessions
	ferme d'appareils : ListRuns
	ferme d'appareils : ListSamples
	ferme d'appareils : ListSuites
	ferme d'appareils : ListTestGridProjects
	ferme d'appareils : ListTestGridSessionActions
	ferme d'appareils : ListTestGridSessionArtifacts
	ferme d'appareils : ListTestGridSessions
	ferme d'appareils : ListTests
	ferme d'appareils : ListUniqueProblems
	ferme d'appareils : ListUploads
	devicefarm:ListVPCEConfigurations
	ferme d'appareils : PurchaseOffering
	ferme d'appareils : RenewOffering
	ferme d'appareils : ScheduleRun
	ferme d'appareils : StopJob

Préfixe de service	Actions
	ferme d'appareils : StopRemoteAccessSession ferme d'appareils : StopRun ferme d'appareils : UpdateDeviceInstance ferme d'appareils : UpdateDevicePool ferme d'appareils : UpdateInstanceProfile ferme d'appareils : UpdateNetworkProfile ferme d'appareils : UpdateProject ferme d'appareils : UpdateTestGridProject ferme d'appareils : UpdateUpload devicefarm:UpdateVPCEConfiguration

Préfixe de service	Actions
devops-guru	gourou de DevOps : AddNotificationChannel gourou de DevOps : DeleteInsight gourou de DevOps : DescribeAccountHealth gourou de DevOps : DescribeAccountOverview gourou de DevOps : DescribeAnomaly gourou de DevOps : DescribeEventSourcesConfig gourou de DevOps : DescribeFeedback gourou de DevOps : DescribeInsight gourou de DevOps : DescribeOrganizationHealth gourou de DevOps : DescribeOrganizationOverview gourou de DevOps : DescribeOrganizationResourceCollectionHealth gourou de DevOps : DescribeResourceCollectionHealth gourou de DevOps : DescribeServiceIntegration gourou de DevOps : GetCostEstimation gourou de DevOps : GetResourceCollection gourou de DevOps : ListAnomaliesForInsight gourou de DevOps : ListAnomalousLogGroups gourou de DevOps : ListEvents gourou de DevOps : ListInsights gourou de DevOps : ListMonitoredResources

Préfixe de service	Actions
	gourou de DevOps : ListNotificationChannels gourou de DevOps : ListOrganizationInsights gourou de DevOps : ListRecommendations gourou de DevOps : PutFeedback gourou de DevOps : RemoveNotificationChannel gourou de DevOps : SearchInsights gourou de DevOps : SearchOrganizationInsights gourou de DevOps : StartCostEstimation gourou de DevOps : UpdateEventSourcesConfig gourou de DevOps : UpdateResourceCollection gourou de DevOps : UpdateServiceIntegration

Préfixe de service	Actions
directconnect	connexion directe : AcceptDirectConnectGatewayAssociationProposal
	connexion directe : AllocateConnectionOnInterconnect
	connexion directe : AllocateHostedConnection
	connexion directe : AllocatePrivateVirtualInterface
	connexion directe : AllocatePublicVirtualInterface
	connexion directe : AllocateTransitVirtualInterface
	connexion directe : AssociateConnectionWithLag
	connexion directe : AssociateHostedConnection
	connexion directe : AssociateMacSecKey
	connexion directe : AssociateVirtualInterface
	connexion directe : ConfirmConnection
	connexion directe : ConfirmCustomerAgreement
	connexion directe : ConfirmPrivateVirtualInterface
	connexion directe : ConfirmPublicVirtualInterface
	connexion directe : ConfirmTransitVirtualInterface
	directconnect:CreateBGPPeer
	connexion directe : CreateConnection
	connexion directe : CreateDirectConnectGateway
	connexion directe : CreateDirectConnectGatewayAssociation
	connexion directe : CreateDirectConnectGatewayAssociationProposal

Préfixe de service	Actions
	connexion directe : CreateInterconnect
	connexion directe : CreateLag
	connexion directe : CreatePrivateVirtualInterface
	connexion directe : CreatePublicVirtualInterface
	connexion directe : CreateTransitVirtualInterface
	directconnect:DeleteBGPPeer
	connexion directe : DeleteConnection
	connexion directe : DeleteDirectConnectGateway
	connexion directe : DeleteDirectConnectGatewayAssociation
	connexion directe : DeleteDirectConnectGatewayAssociationProposal
	connexion directe : DeleteInterconnect
	connexion directe : DeleteLag
	connexion directe : DeleteVirtualInterface
	connexion directe : DescribeConnectionLoa
	connexion directe : DescribeConnections
	connexion directe : DescribeConnectionsOnInterconnect
	connexion directe : DescribeCustomerMetadata
	connexion directe : DescribeDirectConnectGatewayAssociationProposals
	connexion directe : DescribeDirectConnectGatewayAssociations
	connexion directe : DescribeDirectConnectGatewayAttachments

Préfixe de service	Actions
	connexion directe : DescribeDirectConnectGateways
	connexion directe : DescribeHostedConnections
	connexion directe : DescribeInterconnectLoa
	connexion directe : DescribeInterconnects
	connexion directe : DescribeLags
	connexion directe : DescribeLoa
	connexion directe : DescribeLocations
	connexion directe : DescribeRouterConfiguration
	connexion directe : DescribeVirtualGateways
	connexion directe : DescribeVirtualInterfaces
	connexion directe : DisassociateConnectionFromLag
	connexion directe : DisassociateMacSecKey
	connexion directe : ListVirtualInterfaceTestHistory
	connexion directe : StartBgpFailoverTest
	connexion directe : StopBgpFailoverTest
	connexion directe : UpdateConnection
	connexion directe : UpdateDirectConnectGateway
	connexion directe : UpdateDirectConnectGatewayAssociation
	connexion directe : UpdateLag
	connexion directe : UpdateVirtualInterfaceAttributes

Préfixe de service	Actions
dlm	DLM : CreateLifecyclePolicy DLM : DeleteLifecyclePolicy DLM : GetLifecyclePolicies DLM : GetLifecyclePolicy DLM : UpdateLifecyclePolicy

Préfixe de service	Actions
dms	dms : ApplyPendingMaintenanceAction
	dms : AssociateExtensionPack
	dms : BatchStartRecommendations
	dms : CancelMetadataModelCreation
	dms : CancelReplicationTaskAssessmentRun
	dms : CreateDataProvider
	dms : CreateEndpoint
	dms : CreateEventSubscription
	dms : CreateInstanceProfile
	dms : CreateMigrationProject
	dms : CreateReplicationConfig
	dms : CreateReplicationInstance
	dms : CreateReplicationSubnetGroup
	dms : CreateReplicationTask
	dms : DeleteCertificate
	dms : DeleteConnection
	dms : DeleteDataMigration
	dms : DeleteDataProvider
	dms : DeleteEndpoint
	dms : DeleteEventSubscription
	dms : DeleteFleetAdvisorCollector

Préfixe de service	Actions
	dms : DeleteFleetAdvisorDatabases
	dms : DeleteInstanceProfile
	dms : DeleteMigrationProject
	dms : DeleteReplicationConfig
	dms : DeleteReplicationInstance
	dms : DeleteReplicationSubnetGroup
	dms : DeleteReplicationTask
	dms : DeleteReplicationTaskAssessmentRun
	dms : DescribeAccountAttributes
	dms : DescribeApplicableIndividualAssessments
	dms : DescribeCertificates
	dms : DescribeConnections
	dms : DescribeDataMigrations
	dms : DescribeEndpointSettings
	dms : DescribeEndpointTypes
	dms : DescribeEndpoints
	dms : DescribeEngineVersions
	dms : DescribeEventCategories
	dms : DescribeEventSubscriptions
	dms : DescribeEvents
	dms : DescribeFleetAdvisorCollectors

Préfixe de service	Actions
	dms : DescribeFleetAdvisorDatabases
	dms : DescribeFleetAdvisorLsaAnalysis
	dms : DescribeFleetAdvisorSchemaObjectSummary
	dms : DescribeFleetAdvisorSchemas
	dms : DescribeMetadataModel
	dms : DescribeMetadataModelChildren
	dms : DescribeMetadataModelCreations
	dms : DescribeMetadataModelImports
	dms : DescribeOrderableReplicationInstances
	dms : DescribePendingMaintenanceActions
	dms : DescribeRecommendationLimitations
	dms : DescribeRecommendations
	dms : DescribeRefreshSchemasStatus
	dms : DescribeReplicationConfigs
	dms : DescribeReplicationInstanceTaskLogs
	dms : DescribeReplicationInstances
	dms : DescribeReplicationSubnetGroups
	dms : DescribeReplicationTableStatistics
	dms : DescribeReplicationTaskAssessmentResults
	dms : DescribeReplicationTaskAssessmentRuns
	dms : DescribeReplicationTaskIndividualAssessments

Préfixe de service	Actions
	dms : DescribeReplicationTasks
	dms : DescribeReplications
	dms : DescribeSchemas
	dms : DescribeTableStatistics
	dms : ExportMetadataModelAssessment
	dms : ImportCertificate
	dms : ListDataProviders
	dms : ListExtensionPacks
	dms : ListInstanceProfiles
	dms : ListMetadataModelAssessments
	dms : ListMetadataModelConversions
	dms : ListMetadataModelExports
	dms : ListMigrationProjects
	dms : ModifyDataMigration
	dms : ModifyEndpoint
	dms : ModifyEventSubscription
	dms : ModifyReplicationConfig
	dms : ModifyReplicationInstance
	dms : ModifyReplicationSubnetGroup
	dms : ModifyReplicationTask
	dms : MoveReplicationTask

Préfixe de service	Actions
	dms : RebootReplicationInstance
	dms : RefreshSchemas
	dms : ReloadReplicationTables
	dms : ReloadTables
	dms : RunFleetAdvisorLsaAnalysis
	dms : StartMetadataModelAssessment
	dms : StartMetadataModelConversion
	dms : StartMetadataModelCreation
	dms : StartMetadataModelExportAsScripts
	dms : StartMetadataModelExportToTarget
	dms : StartRecommendations
	dms : StartReplication
	dms : StartReplicationTask
	dms : StartReplicationTaskAssessment
	dms : StopDataMigration
	dms : StopReplicationTask
	dms : TestConnection
	dms : UpdateConversionConfiguration
	dms : UpdateDataProvider
	dms : UpdateInstanceProfile
	dms : UpdateMigrationProject

Préfixe de service	Actions
	dms : UpdateSubscriptionsToEventBridge
docdb-elastic	docdb-elastic : ApplyPendingMaintenanceAction docdb-elastic : CopyClusterSnapshot docdb-elastic : DeleteCluster docdb-elastic : DeleteClusterSnapshot docdb-elastic : GetCluster docdb-elastic : GetClusterSnapshot docdb-elastic : GetPendingMaintenanceAction docdb-elastic : ListClusterSnapshots docdb-elastic : ListClusters docdb-elastic : ListPendingMaintenanceActions docdb-elastic : RestoreClusterFromSnapshot docdb-elastic : StartCluster docdb-elastic : StopCluster docdb-elastic : UpdateCluster

Préfixe de service	Actions
dynamodb	dynamodb : CreateBackup
	dynamodb : CreateGlobalTable
	dynamodb : CreateTable
	dynamodb : DeleteBackup
	dynamodb : DeleteTable
	dynamodb : DescribeBackup
	dynamodb : DescribeContinuousBackups
	dynamodb : DescribeContributorInsights
	dynamodb : DescribeEndpoints
	dynamodb : DescribeExport
	dynamodb : DescribeGlobalTable
	dynamodb : DescribeGlobalTableSettings
	dynamodb : DescribeImport
	dynamodb : DescribeKinesisStreamingDestination
	dynamodb : DescribeLimits
	dynamodb : DescribeStream
	dynamodb : DescribeTable
	dynamodb : DescribeTableReplicaAutoScaling
	dynamodb : DescribeTimeToLive
	dynamodb : DisableKinesisStreamingDestination
	dynamodb : EnableKinesisStreamingDestination

Préfixe de service	Actions
	dynamodb : ExportTableToPointInTime
	dynamodb : GetResourcePolicy
	dynamodb : ImportTable
	dynamodb : ListBackups
	dynamodb : ListContributorInsights
	dynamodb : ListExports
	dynamodb : ListGlobalTables
	dynamodb : ListImports
	dynamodb : ListStreams
	dynamodb : ListTables
	dynamodb : RestoreTableFromBackup
	dynamodb : RestoreTableToPointInTime
	dynamodb : UpdateContinuousBackups
	dynamodb : UpdateContributorInsights
	dynamodb : UpdateGlobalTable
	dynamodb : UpdateGlobalTableSettings
	dynamodb : UpdateKinesisStreamingDestination
	dynamodb : UpdateTable
	dynamodb : UpdateTableReplicaAutoScaling
	dynamodb : UpdateTimeToLive

Préfixe de service	Actions
ebs	ebs : CompleteSnapshot ebs : StartSnapshot

Préfixe de service	Actions
ec2	EC2 : AcceptAddressTransfer
	EC2 : AcceptCapacityReservationBillingOwnership
	EC2 : AcceptReservedInstancesExchangeQuote
	EC2 : AcceptTransitGatewayClientVpnAttachment
	EC2 : AcceptTransitGatewayMulticastDomainAssociations
	EC2 : AcceptTransitGatewayPeeringAttachment
	EC2 : AcceptTransitGatewayVpcAttachment
	EC2 : AcceptVpcEndpointConnections
	EC2 : AcceptVpcPeeringConnection
	EC2 : AdvertiseByoipCidr
	EC2 : AllocateAddress
	EC2 : AllocateHosts
	EC2 : AllocateIpamPoolCidr
	EC2 : ApplySecurityGroupsToClientVpnTargetNetwork
	EC2 : AssignIpv6Addresses
	EC2 : AssignPrivateIpAddresses
	EC2 : AssignPrivateNatGatewayAddress
	EC2 : AssociateAddress
	EC2 : AssociateCapacityReservationBillingOwner
	EC2 : AssociateClientVpnTargetNetwork
	EC2 : AssociateDhcpOptions

Préfixe de service	Actions
	EC2 : AssociateEnclaveCertificateIamRole
	EC2 : AssociateIamInstanceProfile
	EC2 : AssociateInstanceEventWindow
	EC2 : AssociateIpamByoasn
	EC2 : AssociateIpamResourceDiscovery
	EC2 : AssociateNatGatewayAddress
	EC2 : AssociateRouteServer
	EC2 : AssociateRouteTable
	EC2 : AssociateSecurityGroupVpc
	EC2 : AssociateSubnetCidrBlock
	EC2 : AssociateTransitGatewayMulticastDomain
	EC2 : AssociateTransitGatewayPolicyTable
	EC2 : AssociateTransitGatewayRouteTable
	EC2 : AssociateTrunkInterface
	EC2 : AssociateVpcCidrBlock
	EC2 : AttachClassicLinkVpc
	EC2 : AttachImageWatermark
	EC2 : AttachInternetGateway
	EC2 : AttachNetworkInterface
	EC2 : AttachVerifiedAccessTrustProvider
	EC2 : AttachVolume

Préfixe de service	Actions
	EC2 : AttachVpnGateway
	EC2 : AuthorizeClientVpnIngress
	EC2 : AuthorizeSecurityGroupEgress
	EC2 : AuthorizeSecurityGroupIngress
	EC2 : BundleInstance
	EC2 : CancelBundleTask
	EC2 : CancelCapacityReservation
	EC2 : CancelCapacityReservationFleets
	EC2 : CancelConversionTask
	EC2 : CancelDeclarativePoliciesReport
	EC2 : CancelExportTask
	EC2 : CancelImageLaunchPermission
	EC2 : CancelImportTask
	EC2 : CancelReservedInstancesListing
	EC2 : CancelSpotFleetRequests
	EC2 : CancelSpotInstanceRequests
	EC2 : ConfirmProductInstance
	EC2 : CopyFpgaImage
	EC2 : CopyImage
	EC2 : CopySnapshot
	EC2 : CopyVolumes

Préfixe de service	Actions
	EC2 : CreateCapacityManagerDataExport
	EC2 : CreateCapacityReservation
	EC2 : CreateCapacityReservationBySplitting
	EC2 : CreateCapacityReservationCancellationQuote
	EC2 : CreateCapacityReservationFleet
	EC2 : CreateCarrierGateway
	EC2 : CreateClientVpnEndpoint
	EC2 : CreateClientVpnRoute
	EC2 : CreateCoipCidr
	EC2 : CreateCoipPool
	EC2 : CreateCustomerGateway
	EC2 : CreateDefaultSubnet
	EC2 : CreateDefaultVpc
	EC2 : CreateDelegateMacVolumeOwnershipTask
	EC2 : CreateDhcpOptions
	EC2 : CreateEgressOnlyInternetGateway
	EC2 : CreateFleet
	EC2 : CreateFlowLogs
	EC2 : CreateFpgaImage
	EC2 : CreateImage
	EC2 : CreateImageUsageReport

Préfixe de service	Actions
	EC2 : CreateInstanceConnectEndpoint
	EC2 : CreateInstanceEventWindow
	EC2 : CreateInstanceExportTask
	EC2 : CreateInternetGateway
	EC2 : CreateInterruptibleCapacityReservationAllocation
	EC2 : CreateIpam
	EC2 : CreateIpamExternalResourceVerificationToken
	EC2 : CreateIpamPolicy
	EC2 : CreateIpamPool
	EC2 : CreateIpamPrefixListResolver
	EC2 : CreateIpamPrefixListResolverTarget
	EC2 : CreateIpamResourceDiscovery
	EC2 : CreateIpamScope
	EC2 : CreateKeyPair
	EC2 : CreateLaunchTemplateVersion
	EC2 : CreateLocalGatewayRoute
	EC2 : CreateLocalGatewayRouteTable
	EC2 : CreateLocalGatewayRouteTableVirtualInterfaceGroupAssoc iation
	EC2 : CreateLocalGatewayRouteTableVpcAssociation
	EC2 : CreateLocalGatewayVirtualInterface

Préfixe de service	Actions
	EC2 : CreateLocalGatewayVirtualInterfaceGroup
	EC2 : CreateMacSystemIntegrityProtectionModificationTask
	EC2 : CreateManagedPrefixList
	EC2 : CreateNatGateway
	EC2 : CreateNetworkAcl
	EC2 : CreateNetworkAclEntry
	EC2 : CreateNetworkInsightsAccessScope
	EC2 : CreateNetworkInsightsPath
	EC2 : CreateNetworkInterface
	EC2 : CreateNetworkInterfacePermission
	EC2 : CreatePlacementGroup
	EC2 : CreatePublicIpv4Pool
	EC2 : CreateReplaceRootVolumeTask
	EC2 : CreateReservedInstancesListing
	EC2 : CreateRestoreImageTask
	EC2 : CreateRoute
	EC2 : CreateRouteServer
	EC2 : CreateRouteServerEndpoint
	EC2 : CreateRouteServerPeer
	EC2 : CreateRouteTable
	EC2 : CreateSecondaryNetwork

Préfixe de service	Actions
	EC2 : CreateSecondarySubnet
	EC2 : CreateSecurityGroup
	EC2 : CreateSnapshots
	EC2 : CreateSpotDatafeedSubscription
	EC2 : CreateStoreImageTask
	EC2 : CreateSubnet
	EC2 : CreateSubnetCidrReservation
	EC2 : CreateTrafficMirrorFilter
	EC2 : CreateTrafficMirrorFilterRule
	EC2 : CreateTrafficMirrorSession
	EC2 : CreateTrafficMirrorTarget
	EC2 : CreateTransitGateway
	EC2 : CreateTransitGatewayConnect
	EC2 : CreateTransitGatewayConnectPeer
	EC2 : CreateTransitGatewayMeteringPolicy
	EC2 : CreateTransitGatewayMeteringPolicyEntry
	EC2 : CreateTransitGatewayMulticastDomain
	EC2 : CreateTransitGatewayPeeringAttachment
	EC2 : CreateTransitGatewayPolicyTable
	EC2 : CreateTransitGatewayPrefixListReference
	EC2 : CreateTransitGatewayRoute

Préfixe de service	Actions
	EC2 : CreateTransitGatewayRouteTable
	EC2 : CreateTransitGatewayRouteTableAnnouncement
	EC2 : CreateTransitGatewayVpcAttachment
	EC2 : CreateVerifiedAccessEndpoint
	EC2 : CreateVerifiedAccessGroup
	EC2 : CreateVerifiedAccessInstance
	EC2 : CreateVerifiedAccessTrustProvider
	EC2 : CreateVolume
	EC2 : CreateVpc
	EC2 : CreateVpcBlockPublicAccessExclusion
	EC2 : CreateVpcEncryptionControl
	EC2 : CreateVpcEndpoint
	EC2 : CreateVpcEndpointConnectionNotification
	EC2 : CreateVpcEndpointServiceConfiguration
	EC2 : CreateVpcPeeringConnection
	EC2 : CreateVpnConcentrator
	EC2 : CreateVpnConnection
	EC2 : CreateVpnConnectionRoute
	EC2 : CreateVpnGateway
	EC2 : DeleteCapacityManagerDataExport
	EC2 : DeleteCarrierGateway

Préfixe de service	Actions
	EC2 : DeleteClientVpnEndpoint
	EC2 : DeleteClientVpnRoute
	EC2 : DeleteCoipCidr
	EC2 : DeleteCoipPool
	EC2 : DeleteCustomerGateway
	EC2 : DeleteDhcpOptions
	EC2 : DeleteEgressOnlyInternetGateway
	EC2 : DeleteFleets
	EC2 : DeleteFlowLogs
	EC2 : DeleteFpgaImage
	EC2 : DeleteImageUsageReport
	EC2 : DeleteInstanceConnectEndpoint
	EC2 : DeleteInstanceEventWindow
	EC2 : DeleteInternetGateway
	EC2 : DeleteIpam
	EC2 : DeleteIpamExternalResourceVerificationToken
	EC2 : DeleteIpamPolicy
	EC2 : DeleteIpamPool
	EC2 : DeleteIpamPrefixListResolver
	EC2 : DeleteIpamPrefixListResolverTarget
	EC2 : DeleteIpamResourceDiscovery

Préfixe de service	Actions
	EC2 : DeleteIpamScope
	EC2 : DeleteKeyPair
	EC2 : DeleteLaunchTemplate
	EC2 : DeleteLaunchTemplateVersions
	EC2 : DeleteLocalGatewayRoute
	EC2 : DeleteLocalGatewayRouteTable
	EC2 : DeleteLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	EC2 : DeleteLocalGatewayRouteTableVpcAssociation
	EC2 : DeleteLocalGatewayVirtualInterface
	EC2 : DeleteLocalGatewayVirtualInterfaceGroup
	EC2 : DeleteManagedPrefixList
	EC2 : DeleteNatGateway
	EC2 : DeleteNetworkAcl
	EC2 : DeleteNetworkAclEntry
	EC2 : DeleteNetworkInsightsAccessScope
	EC2 : DeleteNetworkInsightsAccessScopeAnalysis
	EC2 : DeleteNetworkInsightsAnalysis
	EC2 : DeleteNetworkInsightsPath
	EC2 : DeleteNetworkInterface
	EC2 : DeleteNetworkInterfacePermission

Préfixe de service	Actions
	EC2 : DeletePlacementGroup
	EC2 : DeletePublicIpv4Pool
	EC2 : DeleteQueuedReservedInstances
	EC2 : DeleteRoute
	EC2 : DeleteRouteServer
	EC2 : DeleteRouteServerEndpoint
	EC2 : DeleteRouteServerPeer
	EC2 : DeleteRouteTable
	EC2 : DeleteSecondaryNetwork
	EC2 : DeleteSecondarySubnet
	EC2 : DeleteSecurityGroup
	EC2 : DeleteSpotDatafeedSubscription
	EC2 : DeleteSubnet
	EC2 : DeleteSubnetCidrReservation
	EC2 : DeleteTrafficMirrorFilter
	EC2 : DeleteTrafficMirrorFilterRule
	EC2 : DeleteTrafficMirrorSession
	EC2 : DeleteTrafficMirrorTarget
	EC2 : DeleteTransitGateway
	EC2 : DeleteTransitGatewayClientVpnAttachment
	EC2 : DeleteTransitGatewayConnect

Préfixe de service	Actions
	EC2 : DeleteTransitGatewayConnectPeer
	EC2 : DeleteTransitGatewayMeteringPolicy
	EC2 : DeleteTransitGatewayMeteringPolicyEntry
	EC2 : DeleteTransitGatewayMulticastDomain
	EC2 : DeleteTransitGatewayPeeringAttachment
	EC2 : DeleteTransitGatewayPolicyTable
	EC2 : DeleteTransitGatewayPrefixListReference
	EC2 : DeleteTransitGatewayRoute
	EC2 : DeleteTransitGatewayRouteTable
	EC2 : DeleteTransitGatewayRouteTableAnnouncement
	EC2 : DeleteTransitGatewayVpcAttachment
	EC2 : DeleteVerifiedAccessEndpoint
	EC2 : DeleteVerifiedAccessGroup
	EC2 : DeleteVerifiedAccessInstance
	EC2 : DeleteVerifiedAccessTrustProvider
	EC2 : DeleteVolume
	EC2 : DeleteVpc
	EC2 : DeleteVpcBlockPublicAccessExclusion
	EC2 : DeleteVpcEncryptionControl
	EC2 : DeleteVpcEndpointConnectionNotifications
	EC2 : DeleteVpcEndpointServiceConfigurations

Préfixe de service	Actions
	EC2 : DeleteVpcEndpoints
	EC2 : DeleteVpcPeeringConnection
	EC2 : DeleteVpnConcentrator
	EC2 : DeleteVpnConnection
	EC2 : DeleteVpnConnectionRoute
	EC2 : DeleteVpnGateway
	EC2 : DeprovisionByoipCidr
	EC2 : DeprovisionIpamByoasn
	EC2 : DeprovisionIpamPoolCidr
	EC2 : DeprovisionPublicIpv4PoolCidr
	EC2 : DeregisterImage
	EC2 : DeregisterInstanceEventNotificationAttributes
	EC2 : DeregisterTransitGatewayMulticastGroupMembers
	EC2 : DeregisterTransitGatewayMulticastGroupSources
	EC2 : DescribeAccountAttributes
	EC2 : DescribeAddressTransfers
	EC2 : DescribeAddresses
	EC2 : DescribeAddressesAttribute
	EC2 : DescribeAggregateIdFormat
	EC2 : DescribeAvailabilityZones
	EC2 : DescribeAwsNetworkPerformanceMetricSubscriptions

Préfixe de service	Actions
	EC2 : DescribeBundleTasks
	EC2 : DescribeByoipCidrs
	EC2 : DescribeCapacityBlockExtensionHistory
	EC2 : DescribeCapacityBlockExtensionOfferings
	EC2 : DescribeCapacityBlockStatus
	EC2 : DescribeCapacityBlocks
	EC2 : DescribeCapacityManagerDataExports
	EC2 : DescribeCapacityReservationBillingRequests
	EC2 : DescribeCapacityReservationCancellationQuotes
	EC2 : DescribeCapacityReservationFleets
	EC2 : DescribeCapacityReservationTopology
	EC2 : DescribeCapacityReservations
	EC2 : DescribeCarrierGateways
	EC2 : DescribeClassicLinkInstances
	EC2 : DescribeClientVpnAuthorizationRules
	EC2 : DescribeClientVpnConnections
	EC2 : DescribeClientVpnEndpoints
	EC2 : DescribeClientVpnRoutes
	EC2 : DescribeClientVpnTargetNetworks
	EC2 : DescribeCoipPools
	EC2 : DescribeConversionTasks

Préfixe de service	Actions
	EC2 : DescribeCustomerGateways
	EC2 : DescribeDeclarativePoliciesReports
	EC2 : DescribeDhcpOptions
	EC2 : DescribeEgressOnlyInternetGateways
	EC2 : DescribeElasticGpus
	EC2 : DescribeExportImageTasks
	EC2 : DescribeExportTasks
	EC2 : DescribeFastLaunchImages
	EC2 : DescribeFastSnapshotRestores
	EC2 : DescribeFleetHistory
	EC2 : DescribeFleetInstances
	EC2 : DescribeFleets
	EC2 : DescribeFlowLogs
	EC2 : DescribeFpgaImageAttribute
	EC2 : DescribeFpgaImages
	EC2 : DescribeHostReservationOfferings
	EC2 : DescribeHostReservations
	EC2 : DescribeHosts
	EC2 : DescribeIamInstanceProfileAssociations
	EC2 : DescribeIdFormat
	EC2 : DescribeIdentityIdFormat

Préfixe de service	Actions
	EC2 : DescribeImageAttribute
	EC2 : DescribeImageReferences
	EC2 : DescribeImageUsageReportEntries
	EC2 : DescribeImageUsageReports
	EC2 : DescribeImportImageTasks
	EC2 : DescribeImportSnapshotTasks
	EC2 : DescribeInstanceConnectEndpoints
	EC2 : DescribeInstanceCreditSpecifications
	EC2 : DescribeInstanceEventNotificationAttributes
	EC2 : DescribeInstanceEventWindows
	EC2 : DescribeInstanceImageMetadata
	EC2 : DescribeInstanceSqlHaHistoryStates
	EC2 : DescribeInstanceSqlHaStates
	EC2 : DescribeInstanceTopology
	EC2 : DescribeInstanceTypes
	EC2 : DescribeInternetGateways
	EC2 : DescribeIpamByoasn
	EC2 : DescribeIpamExternalResourceVerificationTokens
	EC2 : DescribeIpamPolicies
	EC2 : DescribeIpamPoolAllocations
	EC2 : DescribeIpamPools

Préfixe de service	Actions
	EC2 : DescribeIpamPrefixListResolverTargets
	EC2 : DescribeIpamPrefixListResolvers
	EC2 : DescribeIpamResourceDiscoveries
	EC2 : DescribeIpamResourceDiscoveryAssociations
	EC2 : DescribeIpamScopes
	EC2 : DescribeIpams
	EC2 : DescribeIpv6Pools
	EC2 : DescribeKeyPairs
	EC2 : DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations
	EC2 : DescribeLocalGatewayRouteTableVpcAssociations
	EC2 : DescribeLocalGatewayRouteTables
	EC2 : DescribeLocalGatewayVirtualInterfaceGroups
	EC2 : DescribeLocalGatewayVirtualInterfaces
	EC2 : DescribeLocalGateways
	EC2 : DescribeLockedSnapshots
	EC2 : DescribeMacHosts
	EC2 : DescribeMacModificationTasks
	EC2 : DescribeManagedPrefixLists
	EC2 : DescribeMovingAddresses
	EC2 : DescribeNatGateways

Préfixe de service	Actions
	EC2 : DescribeNetworkAcls
	EC2 : DescribeNetworkInsightsAccessScopeAnalyses
	EC2 : DescribeNetworkInsightsAccessScopes
	EC2 : DescribeNetworkInsightsAnalyses
	EC2 : DescribeNetworkInsightsPaths
	EC2 : DescribeNetworkInterfaceAttribute
	EC2 : DescribeNetworkInterfacePermissions
	EC2 : DescribeNetworkInterfaces
	EC2 : DescribeOutpostLags
	EC2 : DescribePlacementGroups
	EC2 : DescribePrefixLists
	EC2 : DescribePrincipalIdFormat
	EC2 : DescribePublicIpv4Pools
	EC2 : DescribeRegions
	EC2 : DescribeReplaceRootVolumeTasks
	EC2 : DescribeReservedInstances
	EC2 : DescribeReservedInstancesListings
	EC2 : DescribeReservedInstancesModifications
	EC2 : DescribeReservedInstancesOfferings
	EC2 : DescribeRouteServerEndpoints
	EC2 : DescribeRouteServerPeers

Préfixe de service	Actions
	EC2 : DescribeRouteServers
	EC2 : DescribeRouteTables
	EC2 : DescribeScheduledInstanceAvailability
	EC2 : DescribeScheduledInstances
	EC2 : DescribeSecondaryInterfaces
	EC2 : DescribeSecondaryNetworks
	EC2 : DescribeSecondarySubnets
	EC2 : DescribeSecurityGroupReferences
	EC2 : DescribeSecurityGroupRules
	EC2 : DescribeSecurityGroupVpcAssociations
	EC2 : DescribeSecurityGroups
	EC2 : DescribeServiceLinkVirtualInterfaces
	EC2 : DescribeSnapshotAttribute
	EC2 : DescribeSnapshotTierStatus
	EC2 : DescribeSpotDatafeedSubscription
	EC2 : DescribeSpotFleetInstances
	EC2 : DescribeSpotFleetRequestHistory
	EC2 : DescribeSpotFleetRequests
	EC2 : DescribeSpotInstanceRequests
	EC2 : DescribeSpotPriceHistory
	EC2 : DescribeStaleSecurityGroups

Préfixe de service	Actions
	EC2 : DescribeStoreImageTasks
	EC2 : DescribeTrafficMirrorFilterRules
	EC2 : DescribeTrafficMirrorFilters
	EC2 : DescribeTrafficMirrorSessions
	EC2 : DescribeTrafficMirrorTargets
	EC2 : DescribeTransitGatewayAttachments
	EC2 : DescribeTransitGatewayConnectPeers
	EC2 : DescribeTransitGatewayConnects
	EC2 : DescribeTransitGatewayMeteringPolicies
	EC2 : DescribeTransitGatewayMulticastDomains
	EC2 : DescribeTransitGatewayPeeringAttachments
	EC2 : DescribeTransitGatewayPolicyTables
	EC2 : DescribeTransitGatewayRouteTableAnnouncements
	EC2 : DescribeTransitGatewayRouteTables
	EC2 : DescribeTransitGatewayVpcAttachments
	EC2 : DescribeTransitGateways
	EC2 : DescribeTrunkInterfaceAssociations
	EC2 : DescribeVerifiedAccessEndpoints
	EC2 : DescribeVerifiedAccessGroups
	EC2 : DescribeVerifiedAccessInstanceLoggingConfigurations
	EC2 : DescribeVerifiedAccessInstances

Préfixe de service	Actions
	EC2 : DescribeVerifiedAccessTrustProviders
	EC2 : DescribeVolumeAttribute
	EC2 : DescribeVolumeStatus
	EC2 : DescribeVolumes
	EC2 : DescribeVolumesModifications
	EC2 : DescribeVpcAttribute
	EC2 : DescribeVpcBlockPublicAccessExclusions
	EC2 : DescribeVpcBlockPublicAccessOptions
	EC2 : DescribeVpcClassicLink
	EC2 : DescribeVpcClassicLinkDnsSupport
	EC2 : DescribeVpcEncryptionControls
	EC2 : DescribeVpcEndpointAssociations
	EC2 : DescribeVpcEndpointConnectionNotifications
	EC2 : DescribeVpcEndpointConnections
	EC2 : DescribeVpcEndpointServiceConfigurations
	EC2 : DescribeVpcEndpointServicePermissions
	EC2 : DescribeVpcEndpointServices
	EC2 : DescribeVpcEndpoints
	EC2 : DescribeVpcPeeringConnections
	EC2 : DescribeVpcs
	EC2 : DescribeVpnConcentrators

Préfixe de service	Actions
	EC2 : DescribeVpnConnections
	EC2 : DescribeVpnGateways
	EC2 : DetachClassicLinkVpc
	EC2 : DetachImageWatermark
	EC2 : DetachInternetGateway
	EC2 : DetachNetworkInterface
	EC2 : DetachVerifiedAccessTrustProvider
	EC2 : DetachVolume
	EC2 : DetachVpnGateway
	EC2 : DisableAddressTransfer
	EC2 : DisableAllowedImagesSettings
	EC2 : DisableAwsNetworkPerformanceMetricSubscription
	EC2 : DisableCapacityManager
	EC2 : DisableEbsEncryptionByDefault
	EC2 : DisableFastLaunch
	EC2 : DisableFastSnapshotRestores
	EC2 : DisableImage
	EC2 : DisableImageBlockPublicAccess
	EC2 : DisableImageDeprecation
	EC2 : DisableImageDeregistrationProtection
	EC2 : DisableInstanceSqlHaStandbyDetections

Préfixe de service	Actions
	EC2 : DisableIpamOrganizationAdminAccount
	EC2 : DisableIpamPolicy
	EC2 : DisableRouteServerPropagation
	EC2 : DisableSerialConsoleAccess
	EC2 : DisableSnapshotBlockPublicAccess
	EC2 : DisableTransitGatewayRouteTablePropagation
	EC2 : DisableVgwRoutePropagation
	EC2 : DisableVpcClassicLink
	EC2 : DisableVpcClassicLinkDnsSupport
	EC2 : DisassociateAddress
	EC2 : DisassociateCapacityReservationBillingOwner
	EC2 : DisassociateClientVpnTargetNetwork
	EC2 : DisassociateEnclaveCertificateIamRole
	EC2 : DisassociateIamInstanceProfile
	EC2 : DisassociateInstanceEventWindow
	EC2 : DisassociateIpamByoasn
	EC2 : DisassociateIpamResourceDiscovery
	EC2 : DisassociateNatGatewayAddress
	EC2 : DisassociateRouteServer
	EC2 : DisassociateRouteTable
	EC2 : DisassociateSecurityGroupVpc

Préfixe de service	Actions
	EC2 : DisassociateSubnetCidrBlock
	EC2 : DisassociateTransitGatewayMulticastDomain
	EC2 : DisassociateTransitGatewayPolicyTable
	EC2 : DisassociateTransitGatewayRouteTable
	EC2 : DisassociateTrunkInterface
	EC2 : DisassociateVpcCidrBlock
	EC2 : EnableAddressTransfer
	EC2 : EnableAllowedImagesSettings
	EC2 : EnableAwsNetworkPerformanceMetricSubscription
	EC2 : EnableCapacityManager
	EC2 : EnableEbsEncryptionByDefault
	EC2 : EnableFastLaunch
	EC2 : EnableFastSnapshotRestores
	EC2 : EnableImage
	EC2 : EnableImageBlockPublicAccess
	EC2 : EnableImageDeprecation
	EC2 : EnableImageDeregistrationProtection
	EC2 : EnableInstanceSqlHaStandbyDetections
	EC2 : EnableIpamOrganizationAdminAccount
	EC2 : EnableIpamPolicy
	EC2 : EnableReachabilityAnalyzerOrganizationSharing

Préfixe de service	Actions
	EC2 : EnableRouteServerPropagation
	EC2 : EnableSerialConsoleAccess
	EC2 : EnableSnapshotBlockPublicAccess
	EC2 : EnableTransitGatewayRouteTablePropagation
	EC2 : EnableVgwRoutePropagation
	ec2 : E/S EnableVolume
	EC2 : EnableVpcClassicLink
	EC2 : EnableVpcClassicLinkDnsSupport
	EC2 : ExportClientVpnClientCertificateRevocationList
	EC2 : ExportClientVpnClientConfiguration
	EC2 : ExportImage
	EC2 : ExportTransitGatewayRoutes
	EC2 : ExportVerifiedAccessInstanceClientConfiguration
	EC2 : GetActiveVpnTunnelStatus
	EC2 : GetAllowedImagesSettings
	EC2 : GetAssociatedEnclaveCertificateIamRoles
	EC2 : GetAssociatedIpv6PoolCidrs
	EC2 : GetAwsNetworkPerformanceData
	EC2 : GetCapacityManagerAttributes
	EC2 : GetCapacityManagerMetricData
	EC2 : GetCapacityManagerMetricDimensions

Préfixe de service	Actions
	EC2 : GetCapacityReservationUsage
	EC2 : GetCoipPoolUsage
	EC2 : GetConsoleOutput
	EC2 : GetConsoleScreenshot
	EC2 : GetDeclarativePoliciesReportSummary
	EC2 : GetDefaultCreditSpecification
	EC2 : GetEbsDefaultKmsKeyId
	EC2 : GetEbsEncryptionByDefault
	EC2 : GetEnabledIpamPolicy
	EC2 : GetFlowLogsIntegrationTemplate
	EC2 : GetGroupsForCapacityReservation
	EC2 : GetHostReservationPurchasePreview
	EC2 : GetImageAncestry
	EC2 : GetImageBlockPublicAccessState
	EC2 : GetInstanceMetadataDefaults
	EC2 : GetInstanceTpmEkPub
	EC2 : GetInstanceTypesFromInstanceRequirements
	EC2 : GetInstanceUefiData
	EC2 : GetIpamAddressHistory
	EC2 : GetIpamDiscoveredAccounts
	EC2 : GetIpamDiscoveredPublicAddresses

Préfixe de service	Actions
	EC2 : GetIpamDiscoveredResourceCidrs
	EC2 : GetIpamPolicyAllocationRules
	EC2 : GetIpamPolicyOrganizationTargets
	EC2 : GetIpamPoolAllocations
	EC2 : GetIpamPoolCidrs
	EC2 : GetIpamPrefixListResolverRules
	EC2 : GetIpamPrefixListResolverVersionEntries
	EC2 : GetIpamPrefixListResolverVersions
	EC2 : GetIpamResourceCidrs
	EC2 : GetLaunchTemplateData
	EC2 : GetManagedPrefixListAssociations
	EC2 : GetManagedPrefixListEntries
	EC2 : GetManagedResourceVisibility
	EC2 : GetNetworkInsightsAccessScopeAnalysisFindings
	EC2 : GetNetworkInsightsAccessScopeContent
	EC2 : GetPasswordData
	EC2 : GetReservedInstancesExchangeQuote
	EC2 : GetRouteServerAssociations
	EC2 : GetRouteServerPropagations
	EC2 : GetRouteServerRoutingDatabase
	EC2 : GetSecurityGroupsForVpc

Préfixe de service	Actions
	EC2 : GetSerialConsoleAccessStatus
	EC2 : GetSnapshotBlockPublicAccessState
	EC2 : GetSpotPlacementScores
	EC2 : GetSubnetCidrReservations
	EC2 : GetTransitGatewayAttachmentPropagations
	EC2 : GetTransitGatewayMeteringPolicyEntries
	EC2 : GetTransitGatewayMulticastDomainAssociations
	EC2 : GetTransitGatewayPolicyTableAssociations
	EC2 : GetTransitGatewayPolicyTableEntries
	EC2 : GetTransitGatewayPrefixListReferences
	EC2 : GetTransitGatewayRouteTableAssociations
	EC2 : GetTransitGatewayRouteTablePropagations
	EC2 : GetVerifiedAccessEndpointPolicy
	EC2 : GetVerifiedAccessEndpointTargets
	EC2 : GetVerifiedAccessGroupPolicy
	EC2 : GetVpcResourcesBlockingEncryptionEnforcement
	EC2 : GetVpnConnectionDeviceSampleConfiguration
	EC2 : GetVpnConnectionDeviceTypes
	EC2 : GetVpnTunnelReplacementStatus
	EC2 : ImportClientVpnClientCertificateRevocationList
	EC2 : ImportImage

Préfixe de service	Actions
	EC2 : ImportInstance
	EC2 : ImportKeyPair
	EC2 : ImportSnapshot
	EC2 : ImportVolume
	EC2 : ListImagesInRecycleBin
	EC2 : ListSnapshotsInRecycleBin
	EC2 : ListVolumesInRecycleBin
	EC2 : LockSnapshot
	EC2 : ModifyAddressAttribute
	EC2 : ModifyAvailabilityZoneGroup
	EC2 : ModifyCapacityReservation
	EC2 : ModifyCapacityReservationFleet
	EC2 : ModifyClientVpnEndpoint
	EC2 : ModifyDefaultCreditSpecification
	EC2 : ModifyEbsDefaultKmsKeyId
	EC2 : ModifyFleet
	EC2 : ModifyFpgaImageAttribute
	EC2 : ModifyHosts
	EC2 : ModifyIdFormat
	EC2 : ModifyIdentityIdFormat
	EC2 : ModifyImageAttribute

Préfixe de service	Actions
	EC2 : ModifyInstanceAttribute
	EC2 : ModifyInstanceCapacityReservationAttributes
	EC2 : ModifyInstanceConnectEndpoint
	EC2 : ModifyInstanceCpuOptions
	EC2 : ModifyInstanceCreditSpecification
	EC2 : ModifyInstanceEventStartTime
	EC2 : ModifyInstanceEventWindow
	EC2 : ModifyInstanceMaintenanceOptions
	EC2 : ModifyInstanceMetadataDefaults
	EC2 : ModifyInstanceMetadataOptions
	EC2 : ModifyInstanceNetworkPerformanceOptions
	EC2 : ModifyInstancePlacement
	EC2 : ModifyIpam
	EC2 : ModifyIpamPolicyAllocationRules
	EC2 : ModifyIpamPool
	EC2 : ModifyIpamPoolAllocation
	EC2 : ModifyIpamPrefixListResolver
	EC2 : ModifyIpamPrefixListResolverTarget
	EC2 : ModifyIpamResourceCidr
	EC2 : ModifyIpamResourceDiscovery
	EC2 : ModifyIpamScope

Préfixe de service	Actions
	EC2 : ModifyLaunchTemplate
	EC2 : ModifyLocalGatewayRoute
	EC2 : ModifyManagedPrefixList
	EC2 : ModifyManagedResourceVisibility
	EC2 : ModifyNetworkInterfaceAttribute
	EC2 : ModifyPrivateDnsNameOptions
	EC2 : ModifyPublicIpDnsNameOptions
	EC2 : ModifyReservedInstances
	EC2 : ModifyRouteServer
	EC2 : ModifySecurityGroupRules
	EC2 : ModifySnapshotAttribute
	EC2 : ModifySnapshotTier
	EC2 : ModifySpotFleetRequest
	EC2 : ModifySubnetAttribute
	EC2 : ModifyTrafficMirrorFilterNetworkServices
	EC2 : ModifyTrafficMirrorFilterRule
	EC2 : ModifyTrafficMirrorSession
	EC2 : ModifyTransitGateway
	EC2 : ModifyTransitGatewayMeteringPolicy
	EC2 : ModifyTransitGatewayPrefixListReference
	EC2 : ModifyTransitGatewayVpcAttachment

Préfixe de service	Actions
	EC2 : ModifyVerifiedAccessEndpoint
	EC2 : ModifyVerifiedAccessEndpointPolicy
	EC2 : ModifyVerifiedAccessGroup
	EC2 : ModifyVerifiedAccessGroupPolicy
	EC2 : ModifyVerifiedAccessInstance
	EC2 : ModifyVerifiedAccessInstanceLoggingConfiguration
	EC2 : ModifyVerifiedAccessTrustProvider
	EC2 : ModifyVolume
	EC2 : ModifyVolumeAttribute
	EC2 : ModifyVpcAttribute
	EC2 : ModifyVpcBlockPublicAccessExclusion
	EC2 : ModifyVpcBlockPublicAccessOptions
	EC2 : ModifyVpcEncryptionControl
	EC2 : ModifyVpcEndpoint
	EC2 : ModifyVpcEndpointConnectionNotification
	EC2 : ModifyVpcEndpointServiceConfiguration
	EC2 : ModifyVpcEndpointServicePayerResponsibility
	EC2 : ModifyVpcEndpointServicePermissions
	EC2 : ModifyVpcPeeringConnectionOptions
	EC2 : ModifyVpcTenancy
	EC2 : ModifyVpnConnection

Préfixe de service	Actions
	EC2 : ModifyVpnConnectionOptions
	EC2 : ModifyVpnTunnelCertificate
	EC2 : ModifyVpnTunnelOptions
	EC2 : MonitorInstances
	EC2 : MoveAddressToVpc
	EC2 : MoveByoipCidrToIpam
	EC2 : MoveCapacityReservationInstances
	EC2 : ProvisionByoipCidr
	EC2 : ProvisionIpamByoasn
	EC2 : ProvisionIpamPoolCidr
	EC2 : ProvisionPublicIpv4PoolCidr
	EC2 : PurchaseCapacityBlockExtension
	EC2 : PurchaseHostReservation
	EC2 : PurchaseReservedInstancesOffering
	EC2 : PurchaseScheduledInstances
	EC2 : RebootInstances
	EC2 : RegisterImage
	EC2 : RegisterInstanceEventNotificationAttributes
	EC2 : RegisterTransitGatewayMulticastGroupMembers
	EC2 : RegisterTransitGatewayMulticastGroupSources
	EC2 : RejectCapacityReservationBillingOwnership

Préfixe de service	Actions
	EC2 : RejectTransitGatewayClientVpnAttachment
	EC2 : RejectTransitGatewayMulticastDomainAssociations
	EC2 : RejectTransitGatewayPeeringAttachment
	EC2 : RejectTransitGatewayVpcAttachment
	EC2 : RejectVpcEndpointConnections
	EC2 : RejectVpcPeeringConnection
	EC2 : ReleaseAddress
	EC2 : ReleaseHosts
	EC2 : ReleaseIpamPoolAllocation
	EC2 : ReplaceIamInstanceProfileAssociation
	EC2 : ReplaceImageCriteriaInAllowedImagesSettings
	EC2 : ReplaceNetworkAclAssociation
	EC2 : ReplaceNetworkAclEntry
	EC2 : ReplaceRoute
	EC2 : ReplaceRouteTableAssociation
	EC2 : ReplaceTransitGatewayRoute
	EC2 : ReplaceVpnTunnel
	EC2 : ReportInstanceStatus
	EC2 : RequestSpotFleet
	EC2 : RequestSpotInstances
	EC2 : ResetAddressAttribute

Préfixe de service	Actions
	EC2 : ResetEbsDefaultKmsKeyId
	EC2 : ResetFpgaImageAttribute
	EC2 : ResetImageAttribute
	EC2 : ResetInstanceAttribute
	EC2 : ResetNetworkInterfaceAttribute
	EC2 : ResetSnapshotAttribute
	EC2 : RestoreAddressToClassic
	EC2 : RestoreImageFromRecycleBin
	EC2 : RestoreManagedPrefixListVersion
	EC2 : RestoreSnapshotFromRecycleBin
	EC2 : RestoreSnapshotTier
	EC2 : RestoreVolumeFromRecycleBin
	EC2 : RevokeClientVpnIngress
	EC2 : RevokeSecurityGroupEgress
	EC2 : RevokeSecurityGroupIngress
	EC2 : RunInstances
	EC2 : RunScheduledInstances
	EC2 : SearchLocalGatewayRoutes
	EC2 : SearchTransitGatewayMulticastGroups
	EC2 : SearchTransitGatewayRoutes
	EC2 : SendDiagnosticInterrupt

Préfixe de service	Actions
	EC2 : StartDeclarativePoliciesReport
	EC2 : StartInstances
	EC2 : StartNetworkInsightsAccessScopeAnalysis
	EC2 : StartNetworkInsightsAnalysis
	EC2 : StartVpcEndpointServicePrivateDnsVerification
	EC2 : TerminateClientVpnConnections
	EC2 : UnassignIpv6Addresses
	EC2 : UnassignPrivateIpAddresses
	EC2 : UnassignPrivateNatGatewayAddress
	EC2 : UnlockSnapshot
	EC2 : UnmonitorInstances
	EC2 : UpdateCapacityManagerOrganizationsAccess
	EC2 : UpdateInterruptibleCapacityReservationAllocation
	EC2 : UpdateSecurityGroupRuleDescriptionsEgress
	EC2 : UpdateSecurityGroupRuleDescriptionsIngress
	EC2 : WithdrawByoipCidr

Préfixe de service	Actions
ecr	ecr : BatchCheckLayerAvailability ecr : BatchDeleteImage ecr : BatchGetImage ecr : BatchGetRepositoryScanningConfiguration ecr : CompleteLayerUpload ecr : CreatePullThroughCacheRule ecr : CreateRepositoryCreationTemplate ecr : DeleteLifecyclePolicy ecr : DeletePullThroughCacheRule ecr : DeleteRegistryPolicy ecr : DeleteRepository ecr : DeleteRepositoryCreationTemplate ecr : DeleteRepositoryPolicy ecr : DeleteSigningConfiguration ecr : DescribeImageReplicationStatus ecr : DescribeImageScanFindings ecr : DescribeImages ecr : DescribePullThroughCacheRules ecr : DescribeRegistry ecr : DescribeRepositories ecr : DescribeRepositoryCreationTemplates

Préfixe de service	Actions
	ecr : GetAccountSetting ecr : GetAuthorizationToken ecr : GetDownloadUrlForLayer ecr : GetLifecyclePolicy ecr : GetLifecyclePolicyPreview ecr : GetRegistryPolicy ecr : GetRegistryScanningConfiguration ecr : GetRepositoryPolicy ecr : GetSigningConfiguration ecr : InitiateLayerUpload ecr : ListImages ecr : ListPullTimeUpdateExclusions ecr : PutAccountSetting ecr : PutImage ecr : PutImageScanningConfiguration ecr : PutRegistryPolicy ecr : PutRegistryScanningConfiguration ecr : PutReplicationConfiguration ecr : StartImageScan ecr : StartLifecyclePolicyPreview ecr : UpdatePullThroughCacheRule

Préfixe de service	Actions
	ecr : UpdateRepositoryCreationTemplate ecr : UploadLayerPart ecr : ValidatePullThroughCacheRule

Préfixe de service	Actions
ecr-public	ecr-public : BatchCheckLayerAvailability
	ecr-public : BatchDeleteImage
	ecr-public : CompleteLayerUpload
	ecr-public : CreateRepository
	ecr-public : DeleteRepository
	ecr-public : DeleteRepositoryPolicy
	ecr-public : DescribeImages
	ecr-public : DescribeRegistries
	ecr-public : DescribeRepositories
	ecr-public : GetAuthorizationToken
	ecr-public : GetRegistryCatalogData
	ecr-public : GetRepositoryCatalogData
	ecr-public : GetRepositoryPolicy
	ecr-public : InitiateLayerUpload
	ecr-public : PutImage
	ecr-public : PutRegistryCatalogData
	ecr-public : PutRepositoryCatalogData
	ecr-public : SetRepositoryPolicy
	ecr-public : UploadLayerPart

Préfixe de service	Actions
ecs	ecs : ContinueServiceDeployment ecs : CreateCapacityProvider ecs : CreateCluster ecs : CreateDaemon ecs : CreateService ecs : CreateTaskSet ecs : DeleteAccountSetting ecs : DeleteAttributes ecs : DeleteCapacityProvider ecs : DeleteCluster ecs : DeleteDaemon ecs : DeleteDaemonTaskDefinition ecs : DeleteExpressGatewayService ecs : DeleteService ecs : DeleteTaskDefinitions ecs : DeleteTaskSet ecs : DeregisterContainerInstance ecs : DeregisterTaskDefinition ecs : DescribeCapacityProviders ecs : DescribeClusters ecs : DescribeContainerInstances

Préfixe de service	Actions
	ecs : DescribeDaemon ecs : DescribeDaemonDeployments ecs : DescribeDaemonRevisions ecs : DescribeDaemonTaskDefinition ecs : DescribeExpressGatewayService ecs : DescribeServiceDeployments ecs : DescribeServiceRevisions ecs : DescribeServices ecs : DescribeTaskDefinition ecs : DescribeTaskSets ecs : DescribeTasks ecs : DiscoverPollEndpoint ecs : ExecuteCommand ecs : GetTaskProtection ecs : ListAccountSettings ecs : ListAttributes ecs : ListClusters ecs : ListContainerInstances ecs : ListDaemonDeployments ecs : ListDaemonTaskDefinitions ecs : ListDaemons

Préfixe de service	Actions
	ecs : ListServiceDeployments ecs : ListServices ecs : ListServicesByNamespace ecs : ListTaskDefinitionFamilies ecs : ListTaskDefinitions ecs : ListTasks ecs : PutAccountSetting ecs : PutAccountSettingDefault ecs : PutAttributes ecs : PutClusterCapacityProviders ecs : RegisterContainerInstance ecs : RegisterDaemonTaskDefinition ecs : RunTask ecs : StartTask ecs : StopServiceDeployment ecs : StopTask ecs : SubmitAttachmentStateChanges ecs : SubmitContainerStateChange ecs : SubmitTaskStateChange ecs : UpdateCapacityProvider ecs : UpdateCluster

Préfixe de service	Actions
	ecs : UpdateClusterSettings ecs : UpdateContainerAgent ecs : UpdateContainerInstancesState ecs : UpdateDaemon ecs : UpdateExpressGatewayService ecs : UpdateService ecs : UpdateServicePrimaryTaskSet ecs : UpdateTaskProtection ecs : UpdateTaskSet

Préfixe de service	Actions
eks	Eks : AssociateAccessPolicy
	Eks : AssociateEncryptionConfig
	Eks : AssociateIdentityProviderConfig
	Eks : CancelUpdate
	Eks : CreateAccessEntry
	Eks : CreateAddon
	Eks : CreateCluster
	Eks : CreateEksAnywhereSubscription
	Eks : CreateFargateProfile
	Eks : CreateNodegroup
	Eks : DeleteAccessEntry
	Eks : DeleteAddon
	Eks : DeleteCapability
	Eks : DeleteCluster
	Eks : DeleteEksAnywhereSubscription
	Eks : DeleteFargateProfile
	Eks : DeleteNodegroup
	Eks : DeletePodIdentityAssociation
	Eks : DeregisterCluster
	Eks : DescribeAccessEntry
	Eks : DescribeAddon

Préfixe de service	Actions
	Eks : DescribeAddonConfiguration
	Eks : DescribeAddonVersions
	Eks : DescribeCapability
	Eks : DescribeCluster
	Eks : DescribeClusterVersions
	Eks : DescribeEksAnywhereSubscription
	Eks : DescribeFargateProfile
	Eks : DescribeIdentityProviderConfig
	Eks : DescribeInsight
	Eks : DescribeInsightsRefresh
	Eks : DescribeNodegroup
	Eks : DescribePodIdentityAssociation
	Eks : DescribeUpdate
	Eks : DisassociateAccessPolicy
	Eks : DisassociateIdentityProviderConfig
	Eks : ListAccessEntries
	Eks : ListAccessPolicies
	Eks : ListAddons
	Eks : ListAssociatedAccessPolicies
	Eks : ListCapabilities
	Eks : ListClusters

Préfixe de service	Actions
	Eks : ListEksAnywhereSubscriptions
	Eks : ListFargateProfiles
	Eks : ListIdentityProviderConfigs
	Eks : ListInsights
	Eks : ListNodegroups
	Eks : ListPodIdentityAssociations
	Eks : ListUpdates
	Eks : RegisterCluster
	Eks : StartInsightsRefresh
	Eks : UpdateAccessEntry
	Eks : UpdateAddon
	Eks : UpdateCapability
	Eks : UpdateClusterConfig
	Eks : UpdateClusterVersion
	Eks : UpdateEksAnywhereSubscription
	Eks : UpdateNodegroupConfig
	Eks : UpdateNodegroupVersion
	Eks : UpdatePodIdentityAssociation

Préfixe de service	Actions
elasticache	Cache élastique : AuthorizeCacheSecurityGroupIngress Cache élastique : BatchApplyUpdateAction Cache élastique : BatchStopUpdateAction Cache élastique : CompleteMigration Cache élastique : CopyServerlessCacheSnapshot Cache élastique : CopySnapshot Cache élastique : CreateCacheCluster Cache élastique : CreateCacheParameterGroup Cache élastique : CreateCacheSecurityGroup Cache élastique : CreateCacheSubnetGroup Cache élastique : CreateGlobalReplicationGroup Cache élastique : CreateReplicationGroup Cache élastique : CreateServerlessCache Cache élastique : CreateServerlessCacheSnapshot Cache élastique : CreateSnapshot Cache élastique : CreateUser Cache élastique : CreateUserGroup Cache élastique : DecreaseNodeGroupsInGlobalReplicationGroup Cache élastique : DecreaseReplicaCount Cache élastique : DeleteCacheCluster Cache élastique : DeleteCacheParameterGroup

Préfixe de service	Actions
	Cache élastique : DeleteCacheSecurityGroup Cache élastique : DeleteCacheSubnetGroup Cache élastique : DeleteGlobalReplicationGroup Cache élastique : DeleteReplicationGroup Cache élastique : DeleteServerlessCache Cache élastique : DeleteServerlessCacheSnapshot Cache élastique : DeleteSnapshot Cache élastique : DeleteUser Cache élastique : DeleteUserGroup Cache élastique : DescribeCacheClusters Cache élastique : DescribeCacheEngineVersions Cache élastique : DescribeCacheParameterGroups Cache élastique : DescribeCacheParameters Cache élastique : DescribeCacheSecurityGroups Cache élastique : DescribeCacheSubnetGroups Cache élastique : DescribeEngineDefaultParameters Cache élastique : DescribeEvents Cache élastique : DescribeGlobalReplicationGroups Cache élastique : DescribeReplicationGroups Cache élastique : DescribeReservedCacheNodes Cache élastique : DescribeReservedCacheNodesOfferings

Préfixe de service	Actions
	Cache élastique : DescribeServerlessCacheSnapshots Cache élastique : DescribeServerlessCaches Cache élastique : DescribeServiceUpdates Cache élastique : DescribeSnapshots Cache élastique : DescribeUpdateActions Cache élastique : DescribeUserGroups Cache élastique : DescribeUsers Cache élastique : DisassociateGlobalReplicationGroup Cache élastique : ExportServerlessCacheSnapshot Cache élastique : FailoverGlobalReplicationGroup Cache élastique : IncreaseNodeGroupsInGlobalReplicationGroup Cache élastique : IncreaseReplicaCount Cache élastique : ListAllowedNodeTypeModifications Cache élastique : ModifyCacheCluster Cache élastique : ModifyCacheParameterGroup Cache élastique : ModifyCacheSubnetGroup Cache élastique : ModifyGlobalReplicationGroup Cache élastique : ModifyReplicationGroup Cache élastique : ModifyReplicationGroupShardConfiguration Cache élastique : ModifyServerlessCache Cache élastique : ModifyUser

Préfixe de service	Actions
	Cache élastique : ModifyUserGroup Cache élastique : PurchaseReservedCacheNodesOffering Cache élastique : RebalanceSlotsInGlobalReplicationGroup Cache élastique : RebootCacheCluster Cache élastique : ResetCacheParameterGroup Cache élastique : RevokeCacheSecurityGroupIngress Cache élastique : StartMigration Cache élastique : TestFailover Cache élastique : TestMigration

Préfixe de service	Actions
elasticbeanstalk	tige de haricot élastique : AbortEnvironmentUpdate
	tige de haricot élastique : ApplyEnvironmentManagedAction
	tige de haricot élastique : AssociateEnvironmentOperationsRole
	elasticbeanstalk:CheckDNSAvailability
	tige de haricot élastique : ComposeEnvironments
	tige de haricot élastique : CreateApplication
	tige de haricot élastique : CreateApplicationVersion
	tige de haricot élastique : CreateConfigurationTemplate
	tige de haricot élastique : CreateEnvironment
	tige de haricot élastique : CreatePlatformVersion
	tige de haricot élastique : CreateStorageLocation
	tige de haricot élastique : DeleteApplication
	tige de haricot élastique : DeleteApplicationVersion
	tige de haricot élastique : DeleteConfigurationTemplate
	tige de haricot élastique : DeleteEnvironmentConfiguration
	tige de haricot élastique : DeletePlatformVersion
	tige de haricot élastique : DescribeAccountAttributes
	tige de haricot élastique : DescribeApplicationVersions
	tige de haricot élastique : DescribeApplications
	tige de haricot élastique : DescribeConfigurationOptions
	tige de haricot élastique : DescribeConfigurationSettings

Préfixe de service	Actions
	tige de haricot élastique : DescribeEnvironmentHealth
	tige de haricot élastique : DescribeEnvironmentManagedActionHistory
	tige de haricot élastique : DescribeEnvironmentManagedActions
	tige de haricot élastique : DescribeEnvironmentResources
	tige de haricot élastique : DescribeEnvironments
	tige de haricot élastique : DescribeEvents
	tige de haricot élastique : DescribeInstancesHealth
	tige de haricot élastique : DescribePlatformVersion
	tige de haricot élastique : DisassociateEnvironmentOperationsRole
	tige de haricot élastique : ListAvailableSolutionStacks
	tige de haricot élastique : ListPlatformBranches
	tige de haricot élastique : ListPlatformVersions
	tige de haricot élastique : RebuildEnvironment
	tige de haricot élastique : RequestEnvironmentInfo
	tige de haricot élastique : RestartAppServer
	tige de haricot élastique : RetrieveEnvironmentInfo
	tige de haricot élastique : SwapEnvironmentCNAMEs
	tige de haricot élastique : TerminateEnvironment
	tige de haricot élastique : UpdateApplication
	tige de haricot élastique : UpdateApplicationResourceLifecycle

Préfixe de service	Actions
	tige de haricot élastique : UpdateApplicationVersion tige de haricot élastique : UpdateConfigurationTemplate tige de haricot élastique : UpdateEnvironment tige de haricot élastique : ValidateConfigurationSettings

Préfixe de service	Actions
elasticfilesystem	système de fichiers élastique : CreateAccessPoint système de fichiers élastique : CreateFileSystem système de fichiers élastique : CreateMountTarget système de fichiers élastique : CreateReplicationConfiguration système de fichiers élastique : DeleteAccessPoint système de fichiers élastique : DeleteFileSystem système de fichiers élastique : DeleteFileSystemPolicy système de fichiers élastique : DeleteMountTarget système de fichiers élastique : DeleteReplicationConfiguration système de fichiers élastique : DescribeAccessPoints système de fichiers élastique : DescribeAccountPreferences système de fichiers élastique : DescribeBackupPolicy système de fichiers élastique : DescribeFileSystemPolicy système de fichiers élastique : DescribeFileSystems système de fichiers élastique : DescribeLifecycleConfiguration système de fichiers élastique : DescribeMountTargetSecurityGroups système de fichiers élastique : DescribeMountTargets système de fichiers élastique : DescribeReplicationConfigurations système de fichiers élastique : ModifyMountTargetSecurityGroups système de fichiers élastique : PutAccountPreferences système de fichiers élastique : PutBackupPolicy

Préfixe de service	Actions
	système de fichiers élastique : PutFileSystemPolicy système de fichiers élastique : PutLifecycleConfiguration système de fichiers élastique : UpdateFileSystem système de fichiers élastique : UpdateFileSystemProtection

Préfixe de service	Actions
elasticloadbalancing	équilibrage de charge élastique : AddListenerCertificates
	équilibrage de charge élastique : AddTrustStoreRevocations
	équilibrage de charge élastique : ApplySecurityGroupsToLoadBalancer
	équilibrage de charge élastique : AttachLoadBalancerToSubnets
	équilibrage de charge élastique : ConfigureHealthCheck
	équilibrage de charge élastique : CreateAppCookieStickinessPolicy
	équilibrage de charge élastique : CreateLBCookieStickinessPolicy
	équilibrage de charge élastique : CreateListener
	équilibrage de charge élastique : CreateLoadBalancer
	équilibrage de charge élastique : CreateLoadBalancerListeners
	équilibrage de charge élastique : CreateLoadBalancerPolicy
	équilibrage de charge élastique : CreateRule
	équilibrage de charge élastique : CreateTargetGroup
	équilibrage de charge élastique : CreateTrustStore
	équilibrage de charge élastique : DeleteListener
	équilibrage de charge élastique : DeleteLoadBalancer
	équilibrage de charge élastique : DeleteLoadBalancerListeners
	équilibrage de charge élastique : DeleteLoadBalancerPolicy
	équilibrage de charge élastique : DeleteRule
	équilibrage de charge élastique : DeleteSharedTrustStoreAssociation

Préfixe de service	Actions
	équilibrage de charge élastique : DeleteTargetGroup
	équilibrage de charge élastique : DeleteTrustStore
	équilibrage de charge élastique : DeregisterInstancesFromLoad Balancer
	équilibrage de charge élastique : DeregisterTargets
	équilibrage de charge élastique : DescribeAccountLimits
	équilibrage de charge élastique : DescribeCapacityReservation
	équilibrage de charge élastique : DescribeInstanceHealth
	équilibrage de charge élastique : DescribeListenerAttributes
	équilibrage de charge élastique : DescribeListenerCertificates
	équilibrage de charge élastique : DescribeListeners
	équilibrage de charge élastique : DescribeLoadBalancerAttributes
	équilibrage de charge élastique : DescribeLoadBalancerPolicies
	équilibrage de charge élastique : DescribeLoadBalancerPolicyT ypes
	équilibrage de charge élastique : DescribeLoadBalancers
	équilibrage de charge élastique : DescribeRules
	elasticloadbalancing:DescribeSSLPolicies
	équilibrage de charge élastique : DescribeTargetGroupAttributes
	équilibrage de charge élastique : DescribeTargetGroups
	équilibrage de charge élastique : DescribeTargetHealth
	équilibrage de charge élastique : DescribeTrustStoreAssociations

Préfixe de service	Actions
	équilibrage de charge élastique : DescribeTrustStoreRevocations
	équilibrage de charge élastique : DescribeTrustStores
	équilibrage de charge élastique : DetachLoadBalancerFromSubnets
	équilibrage de charge élastique : DisableAvailabilityZonesFor LoadBalancer
	équilibrage de charge élastique : EnableAvailabilityZonesForL oadBalancer
	équilibrage de charge élastique : GetResourcePolicy
	équilibrage de charge élastique : GetTrustStoreCaCertificates Bundle
	équilibrage de charge élastique : GetTrustStoreRevocationContent
	équilibrage de charge élastique : ModifyCapacityReservation
	équilibrage de charge élastique : ModifyIpPools
	équilibrage de charge élastique : ModifyListener
	équilibrage de charge élastique : ModifyLoadBalancerAttributes
	équilibrage de charge élastique : ModifyRule
	équilibrage de charge élastique : ModifyTargetGroup
	équilibrage de charge élastique : ModifyTargetGroupAttributes
	équilibrage de charge élastique : ModifyTrustStore
	équilibrage de charge élastique : RegisterInstancesWithLoadBa lancer
	équilibrage de charge élastique : RegisterTargets
	équilibrage de charge élastique : RemoveListenerCertificates

Préfixe de service	Actions
	équilibrage de charge élastique : RemoveTrustStoreRevocations
	équilibrage de charge élastique : SetIpAddressType
	équilibrage de charge élastique : SetLoadBalancerListenerSSLCertificate
	équilibrage de charge élastique : SetLoadBalancerPoliciesForBackendServer
	équilibrage de charge élastique : SetLoadBalancerPoliciesOfListener
	équilibrage de charge élastique : SetRulePriorities
	équilibrage de charge élastique : SetSecurityGroups
	équilibrage de charge élastique : SetSubnets

Préfixe de service	Actions
emr-containers	conteneurs EMR : CancelJobRun conteneurs EMR : CreateJobTemplate conteneurs EMR : CreateManagedEndpoint conteneurs EMR : CreateSecurityConfiguration conteneurs EMR : CreateVirtualCluster conteneurs EMR : DeleteJobTemplate conteneurs EMR : DeleteManagedEndpoint conteneurs EMR : DeleteSecurityConfiguration conteneurs EMR : DeleteVirtualCluster conteneurs EMR : DescribeJobRun conteneurs EMR : DescribeJobTemplate conteneurs EMR : DescribeManagedEndpoint conteneurs EMR : DescribeSecurityConfiguration conteneurs EMR : DescribeVirtualCluster conteneurs EMR : GetManagedEndpointSessionCredentials conteneurs EMR : ListJobRuns Conteneurs EMR : ListJobTemplates Conteneurs EMR : ListManagedEndpoints Conteneurs EMR : ListSecurityConfigurations Conteneurs EMR : ListVirtualClusters Conteneurs EMR : StartJobRun

Préfixe de service	Actions
emr-serverless	emr sans serveur : CancelJobRun
	emr sans serveur : CreateApplication
	emr sans serveur : DeleteApplication
	emr sans serveur : GetApplication
	emr sans serveur : GetDashboardForJobRun
	emr sans serveur : GetJobRun
	emr sans serveur : GetResourceDashboard
	emr sans serveur : GetSession
	emr sans serveur : GetSessionEndpoint
	emr sans serveur : ListApplications
	emr sans serveur : ListJobRunAttempts
	emr sans serveur : ListJobRuns
	emr sans serveur : ListSessions
	emr sans serveur : StartApplication
	emr sans serveur : StartJobRun
	emr sans serveur : StopApplication
	emr sans serveur : TerminateSession
	emr sans serveur : UpdateApplication

Préfixe de service	Actions
es	Oui : AcceptInboundConnection Oui : AcceptInboundCrossClusterSearchConnection Oui : AssociatePackage Oui : AuthorizeVpcEndpointAccess Oui : CancelElasticsearchServiceSoftwareUpdate Oui : CancelServiceSoftwareUpdate Oui : CreateDomain Oui : CreateElasticsearchDomain Oui : CreateIndex Oui : CreateOutboundConnection Oui : CreateOutboundCrossClusterSearchConnection Oui : CreatePackage Oui : CreateVpcEndpoint Oui : DeleteDomain Oui : DeleteElasticsearchDomain Oui : DeleteElasticsearchServiceRole Oui : DeleteInboundConnection Oui : DeleteInboundCrossClusterSearchConnection Oui : DeleteIndex Oui : DeleteOutboundConnection Oui : DeleteOutboundCrossClusterSearchConnection

Préfixe de service	Actions
	Oui : DeletePackage Oui : DeleteVpcEndpoint Oui : DescribeDomain Oui : DescribeDomainAutoTunes Oui : DescribeDomainChangeProgress Oui : DescribeDomainConfig Oui : DescribeDomainHealth Oui : DescribeDomainNodes Oui : DescribeDomains Oui : DescribeDryRunProgress Oui : DescribeElasticsearchDomain Oui : DescribeElasticsearchDomainConfig Oui : DescribeElasticsearchDomains Oui : DescribeElasticsearchInstanceTypeLimits Oui : DescribeInboundConnections Oui : DescribeInboundCrossClusterSearchConnections Oui : DescribeInstanceTypeLimits Oui : DescribeOutboundConnections Oui : DescribeOutboundCrossClusterSearchConnections Oui : DescribePackages Oui : DescribeReservedElasticsearchInstanceOfferings

Préfixe de service	Actions
	Oui : DescribeReservedElasticsearchInstances Oui : DescribeReservedInstanceOfferings Oui : DescribeReservedInstances Oui : DescribeVpcEndpoints Oui : DissociatePackage Oui : DissociatePackages Oui : GetCompatibleElasticsearchVersions Oui : GetCompatibleVersions Oui : GetDataSource Oui : GetDomainMaintenanceStatus Oui : GetPackageVersionHistory Oui : GetUpgradeHistory Oui : GetUpgradeStatus Oui : ListDataSources Oui : ListDomainNames Oui : ListDomainsForPackage Oui : ListElasticsearchInstanceTypes Oui : ListElasticsearchVersions Oui : ListInsights Oui : ListInstanceTypeDetails Oui : ListPackagesForDomain

Préfixe de service	Actions
	Oui : ListScheduledActions Oui : ListVersions Oui : ListVpcEndpointAccess Oui : ListVpcEndpoints Oui : ListVpcEndpointsForDomain Oui : PurchaseReservedElasticsearchInstanceOffering Oui : PurchaseReservedInstanceOffering Oui : RejectInboundConnection Oui : RejectInboundCrossClusterSearchConnection Oui : RevokeVpcEndpointAccess Oui : RollbackServiceSoftwareUpdate Oui : StartDomainMaintenance Oui : StartElasticsearchServiceSoftwareUpdate Oui : StartServiceSoftwareUpdate Oui : UpdateDataSource Oui : UpdateDomainConfig Oui : UpdateElasticsearchDomainConfig Oui : UpdateIndex Oui : UpdatePackage Oui : UpdatePackageScope Oui : UpdateScheduledAction

Préfixe de service	Actions
	Oui : UpdateVpcEndpoint Oui : UpgradeDomain Oui : UpgradeElasticsearchDomain

Préfixe de service	Actions
events	événements : ActivateEventSource
	événements : CancelReplay
	événements : CreateApiDestination
	événements : CreateArchive
	événements : CreateConnection
	événements : CreateEndpoint
	événements : CreateEventBus
	événements : CreatePartnerEventSource
	événements : DeactivateEventSource
	événements : DeauthorizeConnection
	événements : DeleteApiDestination
	événements : DeleteArchive
	événements : DeleteConnection
	événements : DeleteEndpoint
	événements : DeleteEventBus
	événements : DeletePartnerEventSource
	événements : DeleteRule
	événements : DescribeApiDestination
	événements : DescribeArchive
	événements : DescribeConnection
	événements : DescribeEndpoint

Préfixe de service	Actions
	événements : DescribeEventBus
	événements : DescribeEventSource
	événements : DescribePartnerEventSource
	événements : DescribeReplay
	événements : DescribeRule
	événements : DisableRule
	événements : EnableRule
	événements : ListApiDestinations
	événements : ListArchives
	événements : ListConnections
	événements : ListEndpoints
	événements : ListEventBuses
	événements : ListEventSources
	événements : ListPartnerEventSourceAccounts
	événements : ListPartnerEventSources
	événements : ListReplays
	événements : ListRuleNamesByTarget
	événements : ListRules
	événements : ListTargetsByRule
	événements : PutPermission
	événements : PutRule

Préfixe de service	Actions
	événements : PutTargets
	événements : RemovePermission
	événements : RemoveTargets
	événements : StartReplay
	événements : TestEventPattern
	événements : UpdateApiDestination
	événements : UpdateArchive
	événements : UpdateConnection
	événements : UpdateEndpoint
	événements : UpdateEventBus

Préfixe de service	Actions
finspace	espace fin : CreateEnvironment espace fin : CreateKxChangeset espace fin : CreateKxCluster espace fin : CreateKxDatabase espace fin : CreateKxDataview espace fin : CreateKxEnvironment espace fin : CreateKxScalingGroup espace fin : CreateKxUser espace fin : CreateKxVolume espace fin : CreateUser espace fin : DeleteEnvironment espace fin : DeleteKxCluster espace fin : DeleteKxClusterNode espace fin : DeleteKxDatabase espace fin : DeleteKxDataview espace fin : DeleteKxEnvironment espace fin : DeleteKxScalingGroup espace fin : DeleteKxUser espace fin : DeleteKxVolume espace fin : GetEnvironment espace fin : GetKxChangeset

Préfixe de service	Actions
	espace fin : GetKxCluster espace fin : GetKxConnectionString espace fin : GetKxDatabase espace fin : GetKxDataview espace fin : GetKxEnvironment espace fin : GetKxScalingGroup espace fin : GetKxUser espace fin : GetKxVolume espace fin : GetLoadSampleDataSetGroupIntoEnvironmentStatus espace fin : GetUser espace fin : ListEnvironments espace fin : ListKxChangesets espace fin : ListKxClusterNodes espace fin : ListKxClusters espace fin : ListKxDatabases espace fin : ListKxDataviews espace fin : ListKxEnvironments espace fin : ListKxScalingGroups espace fin : ListKxUsers espace fin : ListKxVolumes espace fin : ListUsers

Préfixe de service	Actions
	espace fin : LoadSampleDataSetGroupIntoEnvironment espace fin : ResetUserPassword espace fin : UpdateEnvironment espace fin : UpdateKxClusterCodeConfiguration espace fin : UpdateKxClusterDatabases espace fin : UpdateKxDatabase espace fin : UpdateKxDataview espace fin : UpdateKxEnvironment espace fin : UpdateKxEnvironmentNetwork espace fin : UpdateKxUser espace fin : UpdateKxVolume espace fin : UpdateUser
firehose	lance à incendie : CreateDeliveryStream lance à incendie : DeleteDeliveryStream lance à incendie : DescribeDeliveryStream lance à incendie : ListDeliveryStreams lance à incendie : StartDeliveryStreamEncryption lance à incendie : StopDeliveryStreamEncryption lance à incendie : UpdateDestination

Préfixe de service	Actions
fis	poisson : CreateExperimentTemplate poisson : CreateTargetAccountConfiguration poisson : DeleteExperimentTemplate poisson : DeleteTargetAccountConfiguration poisson : GetAction poisson : GetExperiment poisson : GetExperimentTargetAccountConfiguration poisson : GetExperimentTemplate poisson : GetSafetyLever poisson : GetTargetAccountConfiguration poisson : GetTargetResourceType poisson : ListActions poisson : ListExperimentResolvedTargets poisson : ListExperimentTargetAccountConfigurations poisson : ListExperimentTemplates poisson : ListExperiments poisson : ListTargetAccountConfigurations poisson : ListTargetResourceTypes poisson : StartExperiment poisson : StopExperiment poisson : UpdateExperimentTemplate

Préfixe de service	Actions
	poisson : UpdateSafetyLeverState poisson : UpdateTargetAccountConfiguration

Préfixe de service	Actions
fms	fms : AssociateAdminAccount
	fms : AssociateThirdPartyFirewall
	fms : BatchAssociateResource
	fms : BatchDisassociateResource
	fms : DeleteAppsList
	fms : DeleteNotificationChannel
	fms : DeletePolicy
	fms : DeleteProtocolsList
	fms : DeleteResourceSet
	fms : DisassociateAdminAccount
	fms : DisassociateThirdPartyFirewall
	fms : GetAdminAccount
	fms : GetAdminScope
	fms : GetAppsList
	fms : GetComplianceDetail
	fms : GetNotificationChannel
	fms : GetPolicy
	fms : GetProtectionStatus
	fms : GetProtocolsList
	fms : GetResourceSet
	fms : GetThirdPartyFirewallAssociationStatus

Préfixe de service	Actions
	fms : GetViolationDetails
	fms : ListAdminAccountsForOrganization
	fms : ListAdminsManagingAccount
	fms : ListAppsLists
	fms : ListComplianceStatus
	fms : ListDiscoveredResources
	fms : ListMemberAccounts
	fms : ListPolicies
	fms : ListProtocolsLists
	fms : ListResourceSetResources
	fms : ListResourceSets
	fms : ListThirdPartyFirewallFirewallPolicies
	fms : PutAdminAccount
	fms : PutAppsList
	fms : PutNotificationChannel
	fms : PutPolicy
	fms : PutProtocolsList
	fms : PutResourceSet

Préfixe de service	Actions
frauddetector	détecteur de fraude : BatchCreateVariable détecteur de fraude : BatchGetVariable détecteur de fraude : CancelBatchImportJob détecteur de fraude : CancelBatchPredictionJob détecteur de fraude : CreateBatchImportJob détecteur de fraude : CreateBatchPredictionJob détecteur de fraude : CreateDetectorVersion détecteur de fraude : CreateList détecteur de fraude : CreateModel détecteur de fraude : CreateModelVersion détecteur de fraude : CreateRule détecteur de fraude : CreateVariable détecteur de fraude : DeleteBatchImportJob détecteur de fraude : DeleteBatchPredictionJob détecteur de fraude : DeleteDetector détecteur de fraude : DeleteDetectorVersion détecteur de fraude : DeleteEntityType détecteur de fraude : DeleteEvent détecteur de fraude : DeleteEventType détecteur de fraude : DeleteEventsByEventType détecteur de fraude : DeleteExternalModel

Préfixe de service	Actions
	détecteur de fraude : DeleteLabel détecteur de fraude : DeleteList détecteur de fraude : DeleteModel détecteur de fraude : DeleteModelVersion détecteur de fraude : DeleteOutcome détecteur de fraude : DeleteRule détecteur de fraude : DeleteVariable détecteur de fraude : DescribeDetector détecteur de fraude : DescribeModelVersions détecteur de fraude : GetBatchImportJobs détecteur de fraude : GetBatchPredictionJobs détecteur de fraude : GetDeleteEventsByEventTypeStatus détecteur de fraude : GetDetectorVersion détecteur de fraude : GetDetectors détecteur de fraude : GetEntityTypes détecteur de fraude : GetEvent détecteur de fraude : GetEventPrediction détecteur de fraude : GetEventPredictionMetadata détecteur de fraude : GetEventTypes détecteur de fraude : GetExternalModels détecteur de fraude : GetKMSEncryptionKey

Préfixe de service	Actions
	détecteur de fraude : GetLabels détecteur de fraude : GetListElements détecteur de fraude : GetListsMetadata détecteur de fraude : GetModelVersion détecteur de fraude : GetModels détecteur de fraude : GetOutcomes détecteur de fraude : GetRules détecteur de fraude : GetVariables détecteur de fraude : ListEventPredictions détecteur de fraude : PutDetector détecteur de fraude : PutEntityType détecteur de fraude : PutEventType détecteur de fraude : PutExternalModel détecteur de fraude : PutKMSEncryptionKey détecteur de fraude : PutLabel détecteur de fraude : PutOutcome détecteur de fraude : SendEvent détecteur de fraude : UpdateDetectorVersion détecteur de fraude : UpdateDetectorVersionMetadata détecteur de fraude : UpdateDetectorVersionStatus détecteur de fraude : UpdateEventLabel

Préfixe de service	Actions
	détecteur de fraude : UpdateList détecteur de fraude : UpdateModel détecteur de fraude : UpdateModelVersion détecteur de fraude : UpdateModelVersionStatus détecteur de fraude : UpdateRuleMetadata détecteur de fraude : UpdateRuleVersion détecteur de fraude : UpdateVariable

Préfixe de service	Actions
fsx	télécopie : AssociateFileSystemAliases
	télécopie : CancelDataRepositoryTask
	télécopie : CopyBackup
	télécopie : CreateDataRepositoryTask
	télécopie : CreateFileCache
	télécopie : CreateFileSystem
	télécopie : CreateFileSystemFromBackup
	télécopie : CreateSnapshot
	télécopie : CreateStorageVirtualMachine
	télécopie : CreateVolume
	télécopie : CreateVolumeFromBackup
	télécopie : DeleteBackup
	télécopie : DeleteFileCache
	télécopie : DeleteFileSystem
	télécopie : DeleteSnapshot
	télécopie : DeleteStorageVirtualMachine
	télécopie : DeleteVolume
	télécopie : DescribeBackups
	télécopie : DescribeDataRepositoryAssociations
	télécopie : DescribeDataRepositoryTasks
	télécopie : DescribeFileCaches

Préfixe de service	Actions
	télécopie : DescribeFileSystemAliases
	télécopie : DescribeFileSystems
	télécopie : DescribeS3AccessPointAttachments
	télécopie : DescribeSharedVpcConfiguration
	télécopie : DescribeSnapshots
	télécopie : DescribeStorageVirtualMachines
	télécopie : DescribeVolumes
	télécopie : DetachAndDeleteS3AccessPoint
	télécopie : DisassociateFileSystemAliases
	télécopie : ReleaseFileSystemNfsV3Locks
	télécopie : RestoreVolumeFromSnapshot
	télécopie : StartMisconfiguredStateRecovery
	télécopie : UpdateDataRepositoryAssociation
	télécopie : UpdateFileCache
	télécopie : UpdateFileSystem
	télécopie : UpdateSharedVpcConfiguration
	télécopie : UpdateSnapshot
	télécopie : UpdateStorageVirtualMachine
	télécopie : UpdateVolume

Préfixe de service	Actions
gamelift	gamelift : AcceptMatch gamelift : ClaimGameServer gamelift : CreateAlias gamelift : CreateBuild gamelift : CreateContainerGroupDefinition gamelift : CreateFleet gamelift : CreateFleetLocations gamelift : CreateGameServerGroup gamelift : CreateGameSession gamelift : CreateGameSessionQueue gamelift : CreateLocation gamelift : CreateMatchmakingConfiguration gamelift : CreateMatchmakingRuleSet gamelift : CreatePlayerSession gamelift : CreatePlayerSessions gamelift : CreateScript gamelift : CreateVpcPeeringAuthorization gamelift : CreateVpcPeeringConnection gamelift : DeleteAlias gamelift : DeleteBuild gamelift : DeleteContainerGroupDefinition

Préfixe de service	Actions
	gamelift : DeleteFleet gamelift : DeleteFleetLocations gamelift : DeleteGameServerGroup gamelift : DeleteGameSessionQueue gamelift : DeleteLocation gamelift : DeleteMatchmakingConfiguration gamelift : DeleteMatchmakingRuleSet gamelift : DeleteScalingPolicy gamelift : DeleteScript gamelift : DeleteVpcPeeringAuthorization gamelift : DeleteVpcPeeringConnection gamelift : DeregisterCompute gamelift : DeregisterGameServer gamelift : DescribeAlias gamelift : DescribeBuild gamelift : DescribeCompute gamelift : DescribeContainerFleet gamelift : DescribeContainerGroupDefinition gamelift : DescribeContainerGroupPortMappings gamelift : DescribeEC2InstanceLimits gamelift : DescribeFleetAttributes

Préfixe de service	Actions
	gamelift : DescribeFleetCapacity gamelift : DescribeFleetEvents gamelift : DescribeFleetLocationAttributes gamelift : DescribeFleetLocationCapacity gamelift : DescribeFleetLocationUtilization gamelift : DescribeFleetPortSettings gamelift : DescribeFleetUtilization gamelift : DescribeGameServer gamelift : DescribeGameServerGroup gamelift : DescribeGameServerInstances gamelift : DescribeGameSessionDetails gamelift : DescribeGameSessionPlacement gamelift : DescribeGameSessionQueues gamelift : DescribeGameSessions gamelift : DescribeInstances gamelift : DescribeMatchmaking gamelift : DescribeMatchmakingConfigurations gamelift : DescribeMatchmakingRuleSets gamelift : DescribePlayerSessions gamelift : DescribeRuntimeConfiguration gamelift : DescribeScalingPolicies

Préfixe de service	Actions
	gamelift : DescribeScript gamelift : DescribeVpcPeeringAuthorizations gamelift : DescribeVpcPeeringConnections gamelift : GetComputeAccess gamelift : GetComputeAuthToken gamelift : GetGameSessionLogUrl gamelift : GetInstanceAccess gamelift : GetPlayerConnectionDetails gamelift : ListAliases gamelift : ListBuilds gamelift : ListCompute gamelift : ListContainerFleets gamelift : ListContainerGroupDefinitionVersions gamelift : ListContainerGroupDefinitions gamelift : ListFleetDeployments gamelift : ListFleets gamelift : ListGameServerGroups gamelift : ListGameServers gamelift : ListLocations gamelift : ListScripts gamelift : PutScalingPolicy

Préfixe de service	Actions
	gamelift : RegisterCompute gamelift : RegisterGameServer gamelift : RequestUploadCredentials gamelift : ResolveAlias gamelift : ResumeGameServerGroup gamelift : SearchGameSessions gamelift : StartFleetActions gamelift : StartGameSessionPlacement gamelift : StartMatchBackfill gamelift : StartMatchmaking gamelift : StopFleetActions gamelift : StopGameSessionPlacement gamelift : StopMatchmaking gamelift : SuspendGameServerGroup gamelift : TerminateGameSession gamelift : UpdateAlias gamelift : UpdateBuild gamelift : UpdateContainerGroupDefinition gamelift : UpdateFleetAttributes gamelift : UpdateFleetCapacity gamelift : UpdateFleetPortSettings

Préfixe de service	Actions
	gamelift : UpdateGameServer gamelift : UpdateGameServerGroup gamelift : UpdateGameSession gamelift : UpdateGameSessionQueue gamelift : UpdateMatchmakingConfiguration gamelift : UpdateRuntimeConfiguration gamelift : UpdateScript gamelift : ValidateMatchmakingRuleSet

Préfixe de service	Actions
geo	géo : AssociateTrackerConsumer
	géo : BatchDeleteDevicePositionHistory
	géo : BatchDeleteGeofence
	géo : BatchEvaluateGeofences
	géo : BatchGetDevicePosition
	géo : BatchPutGeofence
	géo : BatchUpdateDevicePosition
	géo : CalculateRoute
	géo : CalculateRouteMatrix
	géo : CancelJob
	géo : CreateGeofenceCollection
	géo : CreateMap
	géo : CreatePlaceIndex
	géo : CreateRouteCalculator
	géo : CreateTracker
	géo : DeleteGeofenceCollection
	géo : DeleteKey
	géo : DeleteMap
	géo : DeletePlaceIndex
	géo : DeleteRouteCalculator
	géo : DeleteTracker

Préfixe de service	Actions
	géo : DescribeGeofenceCollection
	géo : DescribeKey
	géo : DescribeMap
	géo : DescribePlaceIndex
	géo : DescribeRouteCalculator
	géo : DescribeTracker
	géo : DisassociateTrackerConsumer
	géo : ForecastGeofenceEvents
	géo : GetDevicePosition
	géo : GetDevicePositionHistory
	géo : GetGeofence
	géo : GetJob
	géo : GetMapGlyphs
	géo : GetMapSprites
	géo : GetMapStyleDescriptor
	géo : GetMapTile
	géo : GetPlace
	géo : ListDevicePositions
	géo : ListGeofenceCollections
	géo : ListGeofences
	géo : ListJobs

Préfixe de service	Actions
	géo : ListKeys
	géo : ListMaps
	géo : ListPlaceIndexes
	géo : ListRouteCalculators
	géo : ListTrackerConsumers
	géo : ListTrackers
	géo : PutGeofence
	géo : SearchPlaceIndexForPosition
	géo : SearchPlaceIndexForSuggestions
	géo : SearchPlaceIndexForText
	géo : UpdateGeofenceCollection
	géo : UpdateKey
	géo : UpdateMap
	géo : UpdatePlaceIndex
	géo : UpdateRouteCalculator
	géo : UpdateTracker
	géo : VerifyDevicePosition

Préfixe de service	Actions
glacier	glacier : AbortMultipartUpload glacier : AbortVaultLock glacier : CompleteMultipartUpload glacier : CompleteVaultLock glacier : CreateVault glacier : DeleteArchive glacier : DeleteVault glacier : DeleteVaultAccessPolicy glacier : DeleteVaultNotifications glacier : DescribeJob glacier : DescribeVault glacier : GetDataRetrievalPolicy glacier : GetJobOutput glacier : GetVaultAccessPolicy glacier : GetVaultLock glacier : GetVaultNotifications glacier : InitiateJob glacier : InitiateMultipartUpload glacier : InitiateVaultLock glacier : ListJobs glacier : ListMultipartUploads

Préfixe de service	Actions
	glacier : ListParts glacier : ListProvisionedCapacity glacier : ListVaults glacier : PurchaseProvisionedCapacity glacier : SetDataRetrievalPolicy glacier : SetVaultAccessPolicy glacier : SetVaultNotifications glacier : UploadArchive glacier : UploadMultipartPart

Préfixe de service	Actions
grafana	Grafana : AssociateLicense
	Grafana : CreateWorkspace
	Grafana : CreateWorkspaceApiKey
	Grafana : CreateWorkspaceServiceAccount
	Grafana : CreateWorkspaceServiceAccountToken
	Grafana : DeleteWorkspace
	Grafana : DeleteWorkspaceApiKey
	Grafana : DeleteWorkspaceServiceAccount
	Grafana : DeleteWorkspaceServiceAccountToken
	Grafana : DescribeWorkspace
	Grafana : DescribeWorkspaceAuthentication
	Grafana : DescribeWorkspaceConfiguration
	Grafana : DisassociateLicense
	Grafana : ListPermissions
	Grafana : ListVersions
	Grafana : ListWorkspaceServiceAccountTokens
	Grafana : ListWorkspaceServiceAccounts
	Grafana : ListWorkspaces
	Grafana : UpdatePermissions
	Grafana : UpdateWorkspace
	Grafana : UpdateWorkspaceAuthentication

Préfixe de service	Actions
	Grafana : UpdateWorkspaceConfiguration

Préfixe de service	Actions
greengrass	herbe verte : AssociateRoleToGroup herbe verte : AssociateServiceRoleToAccount herbe verte : BatchAssociateClientDeviceWithCoreDevice herbe verte : BatchDisassociateClientDeviceFromCoreDevice herbe verte : CancelDeployment herbe verte : CreateComponentVersion herbe verte : CreateConnectorDefinition herbe verte : CreateConnectorDefinitionVersion herbe verte : CreateCoreDefinition herbe verte : CreateCoreDefinitionVersion herbe verte : CreateDeployment herbe verte : CreateDeviceDefinition herbe verte : CreateDeviceDefinitionVersion herbe verte : CreateFunctionDefinition herbe verte : CreateFunctionDefinitionVersion herbe verte : CreateGroup herbe verte : CreateGroupCertificateAuthority herbe verte : CreateGroupVersion herbe verte : CreateLoggerDefinition herbe verte : CreateLoggerDefinitionVersion herbe verte : CreateResourceDefinition

Préfixe de service	Actions
	herbe verte : CreateResourceDefinitionVersion herbe verte : CreateSoftwareUpdateJob herbe verte : CreateSubscriptionDefinition herbe verte : CreateSubscriptionDefinitionVersion herbe verte : DeleteComponent herbe verte : DeleteConnectorDefinition herbe verte : DeleteCoreDefinition herbe verte : DeleteCoreDevice herbe verte : DeleteDeployment herbe verte : DeleteDeviceDefinition herbe verte : DeleteFunctionDefinition herbe verte : DeleteGroup herbe verte : DeleteLoggerDefinition herbe verte : DeleteResourceDefinition herbe verte : DeleteSubscriptionDefinition herbe verte : DescribeComponent herbe verte : DisassociateRoleFromGroup herbe verte : DisassociateServiceRoleFromAccount herbe verte : GetAssociatedRole herbe verte : GetBulkDeploymentStatus herbe verte : GetComponent

Préfixe de service	Actions
	herbe verte : GetComponentVersionArtifact herbe verte : GetConnectivityInfo herbe verte : GetConnectorDefinition herbe verte : GetConnectorDefinitionVersion herbe verte : GetCoreDefinition herbe verte : GetCoreDefinitionVersion herbe verte : GetCoreDevice herbe verte : GetDeployment herbe verte : GetDeploymentStatus herbe verte : GetDeviceDefinition herbe verte : GetDeviceDefinitionVersion herbe verte : GetFunctionDefinition herbe verte : GetFunctionDefinitionVersion herbe verte : GetGroup herbe verte : GetGroupCertificateAuthority herbe verte : GetGroupCertificateConfiguration herbe verte : GetGroupVersion herbe verte : GetLoggerDefinition herbe verte : GetLoggerDefinitionVersion herbe verte : GetResourceDefinition herbe verte : GetResourceDefinitionVersion

Préfixe de service	Actions
	herbe verte : GetServiceRoleForAccount herbe verte : GetSubscriptionDefinition herbe verte : GetSubscriptionDefinitionVersion herbe verte : GetThingRuntimeConfiguration herbe verte : ListBulkDeploymentDetailedReports herbe verte : ListBulkDeployments herbe verte : ListClientDevicesAssociatedWithCoreDevice herbe verte : ListComponentVersions herbe verte : ListComponents herbe verte : ListConnectorDefinitionVersions herbe verte : ListConnectorDefinitions herbe verte : ListCoreDefinitionVersions herbe verte : ListCoreDefinitions herbe verte : ListCoreDevices herbe verte : ListDeployments herbe verte : ListDeviceDefinitionVersions herbe verte : ListDeviceDefinitions herbe verte : ListEffectiveDeployments herbe verte : ListFunctionDefinitionVersions herbe verte : ListFunctionDefinitions herbe verte : ListGroupCertificateAuthorities

Préfixe de service	Actions
	herbe verte : ListGroupVersions herbe verte : ListGroups herbe verte : ListInstalledComponents herbe verte : ListLoggerDefinitionVersions herbe verte : ListLoggerDefinitions herbe verte : ListResourceDefinitionVersions herbe verte : ListResourceDefinitions herbe verte : ListSubscriptionDefinitionVersions herbe verte : ListSubscriptionDefinitions herbe verte : ResetDeployments herbe verte : StartBulkDeployment herbe verte : StopBulkDeployment herbe verte : UpdateConnectivityInfo herbe verte : UpdateConnectorDefinition herbe verte : UpdateCoreDefinition herbe verte : UpdateDeviceDefinition herbe verte : UpdateFunctionDefinition herbe verte : UpdateGroup herbe verte : UpdateGroupCertificateConfiguration herbe verte : UpdateLoggerDefinition herbe verte : UpdateResourceDefinition

Préfixe de service	Actions
	herbe verte : UpdateSubscriptionDefinition herbe verte : UpdateThingRuntimeConfiguration

Préfixe de service	Actions
groundstation	station au sol : CancelContact
	station au sol : CreateConfig
	station au sol : CreateDataflowEndpointGroup
	station au sol : CreateDataflowEndpointGroupV2
	station au sol : CreateEphemeris
	station au sol : CreateMissionProfile
	station au sol : DeleteConfig
	station au sol : DeleteDataflowEndpointGroup
	station au sol : DeleteEphemeris
	station au sol : DeleteMissionProfile
	station au sol : DescribeContact
	station au sol : DescribeContactVersion
	station au sol : DescribeEphemeris
	station au sol : GetConfig
	station au sol : GetDataflowEndpointGroup
	station au sol : GetMinuteUsage
	station au sol : GetMissionProfile
	station au sol : GetSatellite
	station au sol : ListAntennas
	station au sol : ListConfigs
	station au sol : ListContacts

Préfixe de service	Actions
	station au sol : ListDataflowEndpointGroups
	station au sol : ListEphemerides
	station au sol : ListGroundStationReservations
	station au sol : ListGroundStations
	station au sol : ListMissionProfiles
	station au sol : ListSatellites
	station au sol : RegisterAgent
	station au sol : ReserveContact
	station au sol : UpdateAgentStatus
	station au sol : UpdateConfig
	station au sol : UpdateEphemeris
	station au sol : UpdateMissionProfile

Préfixe de service	Actions
guardduty	devoir de garde : AcceptAdministratorInvitation devoir de garde : AcceptInvitation devoir de garde : ArchiveFindings devoir de garde : CreateDetector devoir de garde : CreateFilter guardduty:CreateIPSet devoir de garde : CreateInvestigation devoir de garde : CreateMalwareProtectionPlan devoir de garde : CreateMembers devoir de garde : CreatePublishingDestination devoir de garde : CreateSampleFindings devoir de garde : CreateThreatEntitySet devoir de garde : CreateThreatIntelSet devoir de garde : CreateTrustedEntitySet devoir de garde : DeclineInvitations devoir de garde : DeleteDetector devoir de garde : DeleteFilter guardduty:DeleteIPSet devoir de garde : DeleteInvitations devoir de garde : DeleteMalwareProtectionPlan devoir de garde : DeleteMembers

Préfixe de service	Actions
	devoir de garde : DeletePublishingDestination devoir de garde : DeleteThreatEntitySet devoir de garde : DeleteThreatIntelSet devoir de garde : DeleteTrustedEntitySet devoir de garde : DescribeMalwareScans devoir de garde : DescribeOrganizationConfiguration devoir de garde : DescribePublishingDestination devoir de garde : DisableOrganizationAdminAccount devoir de garde : DisassociateFromAdministratorAccount devoir de garde : DisassociateFromMasterAccount devoir de garde : DisassociateMembers devoir de garde : EnableOrganizationAdminAccount devoir de garde : GetAdministratorAccount devoir de garde : GetCoverageStatistics devoir de garde : GetDetector devoir de garde : GetFilter devoir de garde : GetFindings devoir de garde : GetFindingsStatistics guardduty:GetIPSet devoir de garde : GetInvestigation devoir de garde : GetInvitationsCount

Préfixe de service	Actions
	devoir de garde : GetMalwareProtectionPlan devoir de garde : GetMalwareScan devoir de garde : GetMalwareScanSettings devoir de garde : GetMasterAccount devoir de garde : GetMemberDetectors devoir de garde : GetMembers devoir de garde : GetOrganizationStatistics devoir de garde : GetRemainingFreeTrialDays devoir de garde : GetThreatEntitySet devoir de garde : GetThreatIntelSet devoir de garde : GetTrustedEntitySet devoir de garde : GetUsageStatistics devoir de garde : InviteMembers devoir de garde : ListCoverage devoir de garde : ListDetectors devoir de garde : ListFilters devoir de garde : ListFindings guardduty:ListIPSets devoir de garde : ListInvestigations devoir de garde : ListInvitations devoir de garde : ListMalwareProtectionPlans

Préfixe de service	Actions
	devoir de garde : ListMalwareScans devoir de garde : ListMembers devoir de garde : ListOrganizationAdminAccounts devoir de garde : ListPublishingDestinations devoir de garde : ListThreatEntitySets devoir de garde : ListThreatIntelSets devoir de garde : ListTrustedEntitySets devoir de garde : StartMalwareScan devoir de garde : StartMonitoringMembers devoir de garde : StopMonitoringMembers devoir de garde : UnarchiveFindings devoir de garde : UpdateDetector devoir de garde : UpdateFilter devoir de garde : UpdateFindingsFeedback guardduty:UpdateIPSet devoir de garde : UpdateMalwareProtectionPlan devoir de garde : UpdateMalwareScanSettings devoir de garde : UpdateMemberDetectors devoir de garde : UpdateOrganizationConfiguration devoir de garde : UpdatePublishingDestination devoir de garde : UpdateThreatEntitySet

Préfixe de service	Actions
	devoir de garde : UpdateThreatIntelSet devoir de garde : UpdateTrustedEntitySet

Préfixe de service	Actions
healthlake	lac de santé : CancelFHIRExportJobWithDelete
	lac de santé : CreateDataTransformationProfile
	healthlake:CreateFHIRDatastore
	lac de santé : CreateResource
	lac de santé : DeleteDataTransformationProfile
	healthlake:DeleteFHIRDatastore
	lac de santé : DeleteResource
	lac de santé : DescribeDataTransformationJob
	healthlake:DescribeFHIRDatastore
	lac de santé : DescribeFHIRExportJob
	lac de santé : DescribeFHIRExportJobWithGet
	lac de santé : DescribeFHIRImportJob
	lac de santé : GetCapabilities
	lac de santé : GetDataTransformationProfile
	lac de santé : ListDataTransformationJobs
	lac de santé : ListDataTransformationProfileVersions
	lac de santé : ListDataTransformationProfiles
	healthlake:ListFHIRDatastores
	lac de santé : ListFHIRExportJobs
	lac de santé : ListFHIRImportJobs
	lac de santé : PublishDataTransformationProfile

Préfixe de service	Actions
	lac de santé : ReadResource
	lac de santé : SearchEverything
	lac de santé : SearchWithGet
	lac de santé : SearchWithPost
	lac de santé : StartFHIRExportJob
	lac de santé : StartFHIRExportJobWithPost
	lac de santé : StartFHIRImportJob
	lac de santé : UpdateDataTransformationProfile
	Healthlake : mise à jour de la banque de données HIR
	lac de santé : UpdateProfileWithAgent
	lac de santé : UpdateResource

Préfixe de service	Actions
honeycode	code d'identification : BatchCreateTableRows code d'identification : BatchDeleteTableRows code d'identification : BatchUpdateTableRows code d'identification : BatchUpsertTableRows code d'identification : DescribeTableDataImportJob code d'identification : GetScreenData code d'identification : InvokeScreenAutomation code d'identification : ListTableColumns code d'identification : ListTableRows code d'identification : ListTables code d'identification : QueryTableRows code d'identification : StartTableDataImportJob

Préfixe de service	Actions
iam	iam : AddClientIDToOpenIDConnectProvider
	iam : AddRoleToInstanceProfile
	iam : AddUserToGroup
	iam : AttachGroupPolicy
	iam : AttachRolePolicy
	iam : AttachUserPolicy
	iam : ChangePassword
	iam : CreateAccessKey
	iam : CreateAccountAlias
	iam : CreateGroup
	iam : CreateInstanceProfile
	iam : CreateLoginProfile
	iam : CreateOpenIDConnectProvider
	iam : CreatePolicy
	iam : CreatePolicyVersion
	iam : CreateRole
	iam:CreateSAMLProvider
	iam : CreateServiceLinkedRole
	iam : CreateServiceSpecificCredential
	iam : CreateUser
	iam : CreateVirtualMFADevice

Préfixe de service	Actions
	iam:DeactivateMFADevice
	iam : DeleteAccessKey
	iam : DeleteAccountAlias
	iam : DeleteAccountPasswordPolicy
	iam : DeleteCloudFrontPublicKey
	iam : DeleteGroup
	iam : DeleteGroupPolicy
	iam : DeleteInstanceProfile
	iam : DeleteLoginProfile
	iam : DeleteOpenIDConnectProvider
	iam : DeletePolicy
	iam : DeletePolicyVersion
	iam : DeleteRole
	iam : DeleteRolePermissionsBoundary
	iam : DeleteRolePolicy
	iam:DeleteSAMLProvider
	iam : DeleteSSHPublicKey
	iam : DeleteServerCertificate
	iam : DeleteServiceLinkedRole
	iam : DeleteServiceSpecificCredential
	iam : DeleteSigningCertificate

Préfixe de service	Actions
	iam : DeleteUser
	iam : DeleteUserPermissionsBoundary
	iam : DeleteUserPolicy
	iam : DeleteVirtualMFADevice
	iam : DetachGroupPolicy
	iam : DetachRolePolicy
	iam : DetachUserPolicy
	iam : DisableOrganizationsRootCredentialsManagement
	iam : DisableOrganizationsRootSessions
	iam : DisableOutboundWebIdentityFederation
	iam:EnableMFADevice
	iam : EnableOrganizationsRootCredentialsManagement
	iam : EnableOrganizationsRootSessions
	iam : EnableOutboundWebIdentityFederation
	iam : GenerateCredentialReport
	iam : GenerateOrganizationsAccessReport
	iam : GenerateServiceLastAccessedDetails
	iam : GetAccessKeyLastUsed
	iam : GetAccountAuthorizationDetails
	iam : GetAccountEmailAddress
	iam : GetAccountName

Préfixe de service	Actions
	iam : GetAccountPasswordPolicy
	iam : GetAccountSummary
	iam : GetCloudFrontPublicKey
	iam : GetContextKeysForCustomPolicy
	iam : GetContextKeysForPrincipalPolicy
	iam : GetCredentialReport
	iam : GetGroup
	iam : GetGroupPolicy
	iam : GetInstanceProfile
	iam : GetLoginProfile
	iam:GetMFADevice
	iam : GetOpenIDConnectProvider
	iam : GetOrganizationsAccessReport
	iam : GetOutboundWebIdentityFederationInfo
	iam : GetPolicy
	iam : GetPolicyVersion
	iam : GetRole
	iam : GetRolePolicy
	iam:GetSAMLProvider
	iam : GetSSHPublicKey
	iam : GetServerCertificate

Préfixe de service	Actions
	iam : GetServiceLastAccessedDetails
	iam : GetServiceLastAccessedDetailsWithEntities
	iam : GetServiceLinkedRoleDeletionStatus
	iam : GetUser
	iam : GetUserPolicy
	iam : ListAccessKeys
	iam : ListAccountAliases
	iam : ListAttachedGroupPolicies
	iam : ListAttachedRolePolicies
	iam : ListAttachedUserPolicies
	iam : ListCloudFrontPublicKeys
	iam : ListDelegationRequests
	iam : ListEntitiesForPolicy
	iam : ListGroupPolicies
	iam : ListGroups
	iam : ListGroupsForUser
	iam : ListInstanceProfiles
	iam : ListInstanceProfilesForRole
	iam:ListMFADevices
	iam : ListOpenIDConnectProviders
	iam : ListOrganizationsFeatures

Préfixe de service	Actions
	iam : ListPolicies
	iam : ListPoliciesGrantingServiceAccess
	iam : ListPolicyVersions
	iam : ListRolePolicies
	iam : ListRoles
	iam:ListSAMLProviders
	iam : ListSSHPublicKeys
	iam : ListSTSRegionalEndpointsStatus
	iam : ListServerCertificates
	iam : ListServiceSpecificCredentials
	iam : ListSigningCertificates
	iam : ListUserPolicies
	iam : ListUsers
	iam : ListVirtualMFADevices
	iam : PutGroupPolicy
	iam : PutRolePermissionsBoundary
	iam : PutRolePolicy
	iam : PutUserPermissionsBoundary
	iam : PutUserPolicy
	iam : RemoveClientIDFromOpenIDConnectProvider
	iam : RemoveRoleFromInstanceProfile

Préfixe de service	Actions
	iam : RemoveUserFromGroup
	iam : ResetServiceSpecificCredential
	iam:ResyncMFADevice
	iam : SetDefaultPolicyVersion
	iam : SetSTSRegionalEndpointStatus
	iam : SetSecurityTokenServicePreferences
	iam : SimulateCustomPolicy
	iam : SimulatePrincipalPolicy
	iam : UpdateAccessKey
	iam : UpdateAccountEmailAddress
	iam : UpdateAccountName
	iam : UpdateAccountPasswordPolicy
	iam : UpdateAssumeRolePolicy
	iam : UpdateCloudFrontPublicKey
	iam : UpdateGroup
	iam : UpdateLoginProfile
	iam : UpdateOpenIDConnectProviderThumbprint
	iam : UpdateRole
	iam : UpdateRoleDescription
	iam:UpdateSAMLProvider
	iam : UpdateSSHPublicKey

Préfixe de service	Actions
	iam : UpdateServerCertificate iam : UpdateServiceSpecificCredential iam : UpdateSigningCertificate iam : UpdateUser iam : UploadCloudFrontPublicKey iam : UploadSSHPublicKey iam : UploadServerCertificate iam : UploadSigningCertificate

Préfixe de service	Actions
identitystore	boutique d'identité : CreateGroup boutique d'identité : CreateGroupMembership boutique d'identité : CreateUser boutique d'identité : DeleteGroup boutique d'identité : DeleteGroupMembership boutique d'identité : DeleteUser boutique d'identité : DescribeGroup boutique d'identité : DescribeGroupMembership boutique d'identité : DescribeUser boutique d'identité : GetGroupId boutique d'identité : GetGroupMembershipId boutique d'identité : GetUserId boutique d'identité : IsMemberInGroups boutique d'identité : ListGroupMemberships boutique d'identité : ListGroupMembershipsForMember boutique d'identité : ListGroups boutique d'identité : ListUsers boutique d'identité : UpdateGroup boutique d'identité : UpdateUser

Préfixe de service	Actions
imagebuilder	générateur d'images : CancelImageCreation
	générateur d'images : CancelLifecycleExecution
	générateur d'images : CreateComponent
	générateur d'images : CreateContainerRecipe
	générateur d'images : CreateDistributionConfiguration
	générateur d'images : CreateImage
	générateur d'images : CreateImagePipeline
	générateur d'images : CreateImageRecipe
	générateur d'images : CreateInfrastructureConfiguration
	générateur d'images : CreateLifecyclePolicy
	générateur d'images : CreateWorkflow
	générateur d'images : DeleteComponent
	générateur d'images : DeleteContainerRecipe
	générateur d'images : DeleteDistributionConfiguration
	générateur d'images : DeleteImage
	générateur d'images : DeleteImagePipeline
	générateur d'images : DeleteImageRecipe
	générateur d'images : DeleteInfrastructureConfiguration
	générateur d'images : DeleteLifecyclePolicy
	générateur d'images : DeleteWorkflow
	générateur d'images : DistributeImage

Préfixe de service	Actions
	générateur d'images : GetComponentPolicy
	générateur d'images : GetContainerRecipePolicy
	générateur d'images : GetImagePolicy
	générateur d'images : GetImageRecipePolicy
	générateur d'images : GetLifecycleExecution
	générateur d'images : GetLifecyclePolicy
	générateur d'images : GetMarketplaceResource
	générateur d'images : GetWorkflowExecution
	générateur d'images : GetWorkflowStepExecution
	générateur d'images : ImportComponent
	générateur d'images : ImportDiskImage
	générateur d'images : ImportVmImage
	générateur d'images : ListComponentBuildVersions
	générateur d'images : ListComponents
	générateur d'images : ListContainerRecipes
	générateur d'images : ListDistributionConfigurations
	générateur d'images : ListImageBuildVersions
	générateur d'images : ListImagePackages
	générateur d'images : ListImagePipelineImages
	générateur d'images : ListImagePipelines
	générateur d'images : ListImageRecipes

Préfixe de service	Actions
	générateur d'images : ListImageScanFindingAggregations
	générateur d'images : ListImageScanFindings
	générateur d'images : ListImages
	générateur d'images : ListInfrastructureConfigurations
	générateur d'images : ListLifecycleExecutionResources
	générateur d'images : ListLifecycleExecutions
	générateur d'images : ListLifecyclePolicies
	générateur d'images : ListWaitingWorkflowSteps
	générateur d'images : ListWorkflowExecutions
	générateur d'images : ListWorkflowStepExecutions
	générateur d'images : ListWorkflows
	générateur d'images : PutComponentPolicy
	générateur d'images : PutContainerRecipePolicy
	générateur d'images : PutImagePolicy
	générateur d'images : PutImageRecipePolicy
	générateur d'images : RetryImage
	générateur d'images : SendWorkflowStepAction
	générateur d'images : StartImagePipelineExecution
	générateur d'images : StartResourceStateUpdate
	générateur d'images : UpdateDistributionConfiguration
	générateur d'images : UpdateImagePipeline

Préfixe de service	Actions
	générateur d'images : UpdateInfrastructureConfiguration

Préfixe de service	Actions
inspector	inspecteur : AddAttributesToFindings inspecteur : CreateAssessmentTarget inspecteur : CreateAssessmentTemplate inspecteur : CreateExclusionsPreview inspecteur : CreateResourceGroup inspecteur : DeleteAssessmentRun inspecteur : DeleteAssessmentTarget inspecteur : DeleteAssessmentTemplate inspecteur : DescribeAssessmentRuns inspecteur : DescribeAssessmentTargets inspecteur : DescribeAssessmentTemplates inspecteur : DescribeCrossAccountAccessRole inspecteur : DescribeExclusions inspecteur : DescribeFindings inspecteur : DescribeResourceGroups inspecteur : DescribeRulesPackages inspecteur : GetAssessmentReport inspecteur : GetExclusionsPreview inspecteur : GetTelemetryMetadata inspecteur : ListAssessmentRunAgents inspecteur : ListAssessmentRuns

Préfixe de service	Actions
	inspecteur : ListAssessmentTargets inspecteur : ListAssessmentTemplates inspecteur : ListEventSubscriptions inspecteur : ListExclusions inspecteur : ListFindings inspecteur : ListRulesPackages inspecteur : PreviewAgents inspecteur : RegisterCrossAccountAccessRole inspecteur : RemoveAttributesFromFindings inspecteur : StartAssessmentRun inspecteur : StopAssessmentRun inspecteur : SubscribeToEvent inspecteur : UnsubscribeFromEvent inspecteur : UpdateAssessmentTarget

Préfixe de service	Actions
inspector2	inspecteur 2 : AssociateMember
	inspecteur 2 : BatchGetAccountStatus
	inspecteur 2 : BatchGetCodeSnippet
	inspecteur 2 : BatchGetFindingDetails
	inspecteur 2 : BatchGetFreeTrialInfo
	inspecteur 2 : BatchGetMemberEc2DeepInspectionStatus
	inspecteur 2 : BatchUpdateMemberEc2DeepInspectionStatus
	inspecteur 2 : CancelFindingsReport
	inspecteur 2 : CancelSbomExport
	inspecteur 2 : CreateCisScanConfiguration
	inspecteur 2 : CreateCodeSecurityIntegration
	inspecteur 2 : CreateConnector
	inspecteur 2 : CreateFilter
	inspecteur 2 : CreateFindingsReport
	inspecteur 2 : CreateSbomExport
	inspecteur 2 : DeleteCisScanConfiguration
	inspecteur 2 : DeleteCodeSecurityIntegration
	inspecteur 2 : DeleteConnector
	inspecteur 2 : DeleteFilter
	inspecteur 2 : DescribeOrganizationConfiguration
	inspector2:Disable

Préfixe de service	Actions
	inspecteur 2 : DisableDelegatedAdminAccount
	inspecteur 2 : DisassociateMember
	inspector2:Enable
	inspecteur 2 : EnableDelegatedAdminAccount
	inspecteur 2 : GetCisScanReport
	inspecteur 2 : GetCisScanResultDetails
	inspecteur 2 : GetClustersForImage
	inspecteur 2 : GetCodeSecurityIntegration
	inspecteur 2 : GetCodeSecurityScan
	inspecteur 2 : GetConfiguration
	inspecteur 2 : GetDelegatedAdminAccount
	inspecteur 2 : GetEc2DeepInspectionConfiguration
	inspecteur 2 : GetEncryptionKey
	inspecteur 2 : GetFindingsReportStatus
	inspecteur 2 : GetMember
	inspecteur 2 : GetSbomExport
	inspecteur 2 : ListAccountPermissions
	inspecteur 2 : ListCisScanConfigurations
	inspecteur 2 : ListCisScanResultsAggregatedByChecks
	inspecteur 2 : ListCisScanResultsAggregatedByTargetResource
	inspecteur 2 : ListCisScans

Préfixe de service	Actions
	inspecteur 2 : ListCodeSecurityIntegrations
	inspecteur 2 : ListCodeSecurityScanConfigurations
	inspecteur 2 : ListConnectorScanConfigurations
	inspecteur 2 : ListConnectors
	inspecteur 2 : ListCoverage
	inspecteur 2 : ListCoverageStatistics
	inspecteur 2 : ListDelegatedAdminAccounts
	inspecteur 2 : ListFilters
	inspecteur 2 : ListFindingAggregations
	inspecteur 2 : ListFindings
	inspecteur 2 : ListMembers
	inspecteur 2 : ListUsageTotals
	inspecteur 2 : ResetEncryptionKey
	inspecteur 2 : SearchVulnerabilities
	inspecteur 2 : SendCisSessionHealth
	inspecteur 2 : SendCisSessionTelemetry
	inspecteur 2 : StartCisSession
	inspecteur 2 : StartCodeSecurityScan
	inspecteur 2 : StopCisSession
	inspecteur 2 : UpdateCisScanConfiguration
	inspecteur 2 : UpdateCodeSecurityIntegration

Préfixe de service	Actions
	inspecteur 2 : UpdateConfiguration
	inspecteur 2 : UpdateConnector
	inspecteur 2 : UpdateEc2DeepInspectionConfiguration
	inspecteur 2 : UpdateEncryptionKey
	inspecteur 2 : UpdateFilter
	inspecteur 2 : UpdateOrgEc2DeepInspectionConfiguration
	inspecteur 2 : UpdateOrganizationConfiguration

Préfixe de service	Actions
iot	IoT : AcceptCertificateTransfer IoT : AddThingToBillingGroup IoT : AddThingToThingGroup IoT : AssociateSbomWithPackageVersion IoT : AssociateTargetsWithJob IoT : AttachPolicy IoT : AttachPrincipalPolicy IoT : AttachSecurityProfile IoT : AttachThingPrincipal IoT : CancelAuditMitigationActionsTask IoT : CancelAuditTask IoT : CancelCertificateTransfer IoT : CancelDetectMitigationActionsTask IoT : CancelJob IoT : CancelJobExecution IoT : ClearDefaultAuthorizer IoT : ConfirmTopicRuleDestination IoT : CreateAuditSuppression IoT : CreateAuthorizer IoT : CreateBillingGroup IoT : CreateCertificateFromCsr

Préfixe de service	Actions
	IoT : CreateCertificateProvider IoT : CreateCommand IoT : CreateCustomMetric IoT : CreateDimension IoT : CreateDomainConfiguration IoT : CreateDynamicThingGroup IoT : CreateFleetMetric IoT : CreateJob IoT : CreateJobTemplate IoT : CreateKeysAndCertificate IoT : CreateMitigationAction iot:CreateOTAUpdate IoT : CreatePackage IoT : CreatePackageVersion IoT : CreatePolicy IoT : CreatePolicyVersion IoT : CreateProvisioningClaim IoT : CreateProvisioningTemplate IoT : CreateProvisioningTemplateVersion IoT : CreateRoleAlias IoT : CreateScheduledAudit

Préfixe de service	Actions
	IoT : CreateSecurityProfile IoT : CreateStream IoT : CreateThing IoT : CreateThingGroup IoT : CreateThingType IoT : CreateTopicRule IoT : CreateTopicRuleDestination IoT : DeleteAccountAuditConfiguration IoT : DeleteAuditSuppression IoT : DeleteAuthorizer IoT : DeleteBillingGroup iot:DeleteCACertificate IoT : DeleteCertificate IoT : DeleteCertificateProvider IoT : DeleteCommand IoT : DeleteCustomMetric IoT : DeleteDimension IoT : DeleteDomainConfiguration IoT : DeleteDynamicThingGroup IoT : DeleteFleetMetric IoT : DeleteJob

Préfixe de service	Actions
	IoT : DeleteJobExecution
	IoT : DeleteJobTemplate
	IoT : DeleteMitigationAction
	iot:DeleteOTAUpdate
	IoT : DeletePackage
	IoT : DeletePackageVersion
	IoT : DeletePolicy
	IoT : DeletePolicyVersion
	IoT : DeleteProvisioningTemplate
	IoT : DeleteProvisioningTemplateVersion
	IoT : DeleteRegistrationCode
	IoT : DeleteRoleAlias
	IoT : DeleteScheduledAudit
	IoT : DeleteSecurityProfile
	IoT : DeleteStream
	IoT : DeleteThing
	IoT : DeleteThingGroup
	IoT : DeleteThingType
	IoT : DeleteTopicRule
	IoT : DeleteTopicRuleDestination
	IoT : DeleteV2LoggingLevel

Préfixe de service	Actions
	IoT : DeprecateThingType IoT : DescribeAccountAuditConfiguration IoT : DescribeAuditFinding IoT : DescribeAuditMitigationActionsTask IoT : DescribeAuditSuppression IoT : DescribeAuditTask IoT : DescribeAuthorizer IoT : DescribeBillingGroup iot:DescribeCACertificate IoT : DescribeCertificate IoT : DescribeCertificateProvider IoT : DescribeCustomMetric IoT : DescribeDefaultAuthorizer IoT : DescribeDetectMitigationActionsTask IoT : DescribeDimension IoT : DescribeDomainConfiguration IoT : DescribeEncryptionConfiguration IoT : DescribeEndpoint IoT : DescribeEventConfigurations IoT : DescribeFleetMetric IoT : DescribeIndex

Préfixe de service	Actions
	IoT : DescribeJob IoT : DescribeJobExecution IoT : DescribeJobTemplate IoT : DescribeManagedJobTemplate IoT : DescribeMitigationAction IoT : DescribeProvisioningTemplate IoT : DescribeProvisioningTemplateVersion IoT : DescribeRoleAlias IoT : DescribeScheduledAudit IoT : DescribeSecurityProfile IoT : DescribeStream IoT : DescribeThing IoT : DescribeThingGroup IoT : DescribeThingRegistrationTask IoT : DescribeThingType IoT : DetachPolicy IoT : DetachPrincipalPolicy IoT : DetachSecurityProfile IoT : DetachThingPrincipal IoT : DisableTopicRule IoT : DisassociateSbomFromPackageVersion

Préfixe de service	Actions
	IoT : EnableTopicRule IoT : GetBehaviorModelTrainingSummaries IoT : GetBucketsAggregation IoT : GetCardinality IoT : GetCommand IoT : GetEffectivePolicies IoT : GetJobDocument IoT : GetLoggingOptions iot:GetOTAUpdate IoT : GetPackage IoT : GetPackageConfiguration IoT : GetPackageVersion IoT : GetPercentiles IoT : GetPolicy IoT : GetPolicyVersion IoT : GetRegistrationCode IoT : GetStatistics IoT : GetThingConnectivityData IoT : GetTopicRule IoT : GetTopicRuleDestination IoT : GetV2LoggingOptions

Préfixe de service	Actions
	IoT : ListActiveViolations IoT : ListAttachedPolicies IoT : ListAuditFindings IoT : ListAuditMitigationActionsExecutions IoT : ListAuditMitigationActionsTasks IoT : ListAuditSuppressions IoT : ListAuditTasks IoT : ListAuthorizers IoT : ListBillingGroups iot:ListCACertificates IoT : ListCertificateProviders IoT : ListCertificates IoT : ListCertificatesByCA IoT : ListCommands IoT : ListCustomMetrics IoT : ListDetectMitigationActionsExecutions IoT : ListDetectMitigationActionsTasks IoT : ListDimensions IoT : ListDomainConfigurations IoT : ListFleetMetrics IoT : ListIndices

Préfixe de service	Actions
	IoT : ListJobExecutionsForJob IoT : ListJobExecutionsForThing IoT : ListJobTemplates IoT : ListJobs IoT : ListManagedJobTemplates IoT : ListMetricValues IoT : ListMitigationActions iot:ListOTAUpdates IoT : ListOutgoingCertificates IoT : ListPackageVersions IoT : ListPackages IoT : ListPolicies IoT : ListPolicyPrincipals IoT : ListPolicyVersions IoT : ListPrincipalPolicies IoT : ListPrincipalThings IoT : ListPrincipalThingsV2 IoT : ListProvisioningTemplateVersions IoT : ListProvisioningTemplates IoT : ListRelatedResourcesForAuditFinding IoT : ListRoleAliases

Préfixe de service	Actions
	IoT : ListSbomValidationResults
	IoT : ListScheduledAudits
	IoT : ListSecurityProfiles
	IoT : ListSecurityProfilesForTarget
	IoT : ListStreams
	IoT : ListTargetsForPolicy
	IoT : ListTargetsForSecurityProfile
	IoT : ListThingGroups
	IoT : ListThingGroupsForThing
	IoT : ListThingPrincipals
	IoT : ListThingPrincipalsV2
	IoT : ListThingRegistrationTaskReports
	IoT : ListThingRegistrationTasks
	IoT : ListThingTypes
	IoT : ListThings
	IoT : ListThingsInBillingGroup
	IoT : ListThingsInThingGroup
	IoT : ListTopicRuleDestinations
	IoT : ListTopicRules
	IoT : ListV2LoggingLevels
	IoT : ListViolationEvents

Préfixe de service	Actions
	IoT : PutVerificationStateOnViolation iot:RegisterCACertificate IoT : RegisterCertificate IoT : RegisterCertificateWithoutCA IoT : RegisterThing IoT : RejectCertificateTransfer IoT : RemoveThingFromBillingGroup IoT : RemoveThingFromThingGroup IoT : ReplaceTopicRule IoT : SearchIndex IoT : SetDefaultAuthorizer IoT : SetDefaultPolicyVersion IoT : SetLoggingOptions IoT : SetV2LoggingLevel IoT : SetV2LoggingOptions IoT : StartAuditMitigationActionsTask IoT : StartDetectMitigationActionsTask IoT : StartOnDemandAuditTask IoT : StartThingRegistrationTask IoT : StopThingRegistrationTask IoT : TestAuthorization

Préfixe de service	Actions
	IoT : TestInvokeAuthorizer IoT : TransferCertificate IoT : UpdateAccountAuditConfiguration IoT : UpdateAuditSuppression IoT : UpdateAuthorizer IoT : UpdateBillingGroup iot:UpdateCACertificate IoT : UpdateCertificate IoT : UpdateCertificateProvider IoT : UpdateCommand IoT : UpdateCustomMetric IoT : UpdateDimension IoT : UpdateDomainConfiguration IoT : UpdateDynamicThingGroup IoT : UpdateEncryptionConfiguration IoT : UpdateEventConfigurations IoT : UpdateFleetMetric IoT : UpdateIndexingConfiguration IoT : UpdateJob IoT : UpdateMitigationAction IoT : UpdatePackage

Préfixe de service	Actions
	IoT : UpdatePackageConfiguration IoT : UpdatePackageVersion IoT : UpdateProvisioningTemplate IoT : UpdateRoleAlias IoT : UpdateScheduledAudit IoT : UpdateSecurityProfile IoT : UpdateStream IoT : UpdateThing IoT : UpdateThingGroup IoT : UpdateThingGroupsForThing IoT : UpdateThingType IoT : UpdateTopicRuleDestination IoT : ValidateSecurityProfileBehaviors

Préfixe de service	Actions
iotanalytics	Analyse de l'IoT : CancelPipelineReprocessing
	Analyse de l'IoT : CreateChannel
	Analyse de l'IoT : CreateDataset
	Analyse de l'IoT : CreateDatasetContent
	Analyse de l'IoT : CreateDatastore
	Analyse de l'IoT : CreatePipeline
	Analyse de l'IoT : DeleteChannel
	Analyse de l'IoT : DeleteDataset
	Analyse de l'IoT : DeleteDatasetContent
	Analyse de l'IoT : DeleteDatastore
	Analyse de l'IoT : DeletePipeline
	Analyse de l'IoT : DescribeChannel
	Analyse de l'IoT : DescribeDataset
	Analyse de l'IoT : DescribeDatastore
	Analyse de l'IoT : DescribeLoggingOptions
	Analyse de l'IoT : DescribePipeline
	Analyse de l'IoT : GetDatasetContent
	Analyse de l'IoT : ListChannels
	Analyse de l'IoT : ListDatasetContents
	Analyse de l'IoT : ListDatasets
	Analyse de l'IoT : ListDatastores

Préfixe de service	Actions
	Analyse de l'IoT : ListPipelines Analyse de l'IoT : PutLoggingOptions Analyse de l'IoT : RunPipelineActivity Analyse de l'IoT : SampleChannelData Analyse de l'IoT : StartPipelineReprocessing Analyse de l'IoT : UpdateChannel Analyse de l'IoT : UpdateDataset Analyse de l'IoT : UpdateDatastore Analyse de l'IoT : UpdatePipeline
iotdeviceadvisor	conseiller pour appareils IoT : CreateSuiteDefinition conseiller pour appareils IoT : DeleteSuiteDefinition conseiller pour appareils IoT : GetEndpoint conseiller pour appareils IoT : GetSuiteDefinition conseiller pour appareils IoT : GetSuiteRun conseiller pour appareils IoT : GetSuiteRunReport conseiller pour appareils IoT : ListSuiteDefinitions conseiller pour appareils IoT : ListSuiteRuns conseiller pour appareils IoT : StartSuiteRun conseiller pour appareils IoT : StopSuiteRun conseiller pour appareils IoT : UpdateSuiteDefinition

Préfixe de service	Actions
iotevents	événements liés à l'Internet des objets : BatchAcknowledgeAlarm
	événements liés à l'Internet des objets : BatchDeleteDetector
	événements liés à l'Internet des objets : BatchDisableAlarm
	événements liés à l'Internet des objets : BatchEnableAlarm
	événements liés à l'Internet des objets : BatchResetAlarm
	événements liés à l'Internet des objets : BatchSnoozeAlarm
	événements liés à l'Internet des objets : BatchUpdateDetector
	événements liés à l'Internet des objets : CreateAlarmModel
	événements liés à l'Internet des objets : CreateDetectorModel
	événements liés à l'Internet des objets : CreateInput
	événements liés à l'Internet des objets : DeleteAlarmModel
	événements liés à l'Internet des objets : DeleteDetectorModel
	événements liés à l'Internet des objets : DeleteInput
	événements liés à l'Internet des objets : DescribeAlarm
	événements liés à l'Internet des objets : DescribeAlarmModel
	événements liés à l'Internet des objets : DescribeDetector
	événements liés à l'Internet des objets : DescribeDetectorModel
	événements liés à l'Internet des objets : DescribeDetectorModelAnalysis
	événements liés à l'Internet des objets : DescribeInput
	événements liés à l'Internet des objets : DescribeLoggingOptions

Préfixe de service	Actions
	événements liés à l'Internet des objets : GetDetectorModelAnalysisResults événements liés à l'Internet des objets : ListAlarmModelVersions événements liés à l'Internet des objets : ListAlarmModels événements liés à l'Internet des objets : ListAlarms événements liés à l'Internet des objets : ListDetectorModelVersions événements liés à l'Internet des objets : ListDetectorModels événements liés à l'Internet des objets : ListDetectors événements liés à l'Internet des objets : ListInputRoutings événements liés à l'Internet des objets : ListInputs événements liés à l'Internet des objets : PutLoggingOptions événements liés à l'Internet des objets : StartDetectorModelAnalysis événements liés à l'Internet des objets : UpdateAlarmModel événements liés à l'Internet des objets : UpdateDetectorModel événements liés à l'Internet des objets : UpdateInput
iotfleethub	IoT Fleethub : CreateApplication IoT Fleethub : DeleteApplication IoT Fleethub : DescribeApplication IoT Fleethub : ListApplications IoT Fleethub : UpdateApplication

Préfixe de service	Actions
iotsitewise	à l'extérieur du site : AssociateAssets
	à l'extérieur du site : AssociateTimeSeriesToAssetProperty
	à l'extérieur du site : BatchAssociateProjectAssets
	à l'extérieur du site : BatchDisassociateProjectAssets
	à l'extérieur du site : CreateAccessPolicy
	à l'extérieur du site : CreateAsset
	à l'extérieur du site : CreateAssetModel
	à l'extérieur du site : CreateAssetModelCompositeModel
	à l'extérieur du site : CreateBulkImportJob
	à l'extérieur du site : CreateComputationModel
	à l'extérieur du site : CreateDashboard
	à l'extérieur du site : CreateDataset
	à l'extérieur du site : CreateGateway
	à l'extérieur du site : CreatePortal
	à l'extérieur du site : CreateProject
	à l'extérieur du site : DeleteAccessPolicy
	à l'extérieur du site : DeleteAsset
	à l'extérieur du site : DeleteAssetModel
	à l'extérieur du site : DeleteAssetModelCompositeModel
	à l'extérieur du site : DeleteComputationModel
	à l'extérieur du site : DeleteDashboard

Préfixe de service	Actions
	à l'extérieur du site : DeleteDataset
	à l'extérieur du site : DeleteGateway
	à l'extérieur du site : DeletePortal
	à l'extérieur du site : DeleteProject
	à l'extérieur du site : DeleteTimeSeries
	à l'extérieur du site : DescribeAccessPolicy
	à l'extérieur du site : DescribeAsset
	à l'extérieur du site : DescribeAssetCompositeModel
	à l'extérieur du site : DescribeAssetModel
	à l'extérieur du site : DescribeAssetModelCompositeModel
	à l'extérieur du site : DescribeAssetModelInterfaceRelationship
	à l'extérieur du site : DescribeAssetProperty
	à l'extérieur du site : DescribeBulkImportJob
	à l'extérieur du site : DescribeComputationModel
	à l'extérieur du site : DescribeComputationModelExecutionSummary
	à l'extérieur du site : DescribeDashboard
	à l'extérieur du site : DescribeDataset
	à l'extérieur du site : DescribeDefaultEncryptionConfiguration
	à l'extérieur du site : DescribeExecution
	à l'extérieur du site : DescribeGateway
	à l'extérieur du site : DescribeGatewayCapabilityConfiguration

Préfixe de service	Actions
	à l'extérieur du site : DescribeLoggingOptions
	à l'extérieur du site : DescribePortal
	à l'extérieur du site : DescribeProject
	à l'extérieur du site : DescribeStorageConfiguration
	à l'extérieur du site : DescribeTimeSeries
	à l'extérieur du site : DisassociateAssets
	à l'extérieur du site : DisassociateTimeSeriesFromAssetProperty
	à l'extérieur du site : ExecuteAction
	à l'extérieur du site : ExecuteQuery
	à l'extérieur du site : ListAccessPolicies
	à l'extérieur du site : ListActions
	à l'extérieur du site : ListAssetModelCompositeModels
	à l'extérieur du site : ListAssetModelProperties
	à l'extérieur du site : ListAssetModels
	en dehors du site : ListAssetProperties
	en dehors du site : ListAssetRelationships
	en dehors du site : ListAssets
	en dehors du site : ListAssociatedAssets
	en dehors du site : ListBulkImportJobs
	en dehors du site : ListCompositionRelationships
	en dehors du site : ListComputationModelDataBindingUsages

Préfixe de service	Actions
	en dehors du site : ListComputationModelResolveToResources
	en dehors du site : ListComputationModels
	en dehors du site : ListDashboards
	en dehors du site : ListDatasets
	en dehors du site : ListExecutions
	en dehors du site : ListGateways
	en dehors du site : ListInterfaceRelationships
	en dehors du site : ListPortals
	en dehors du site : ListProjectAssets
	en dehors du site : ListProjects
	en dehors du site : ListTimeSeries
	en dehors du site : PutDefaultEncryptionConfiguration
	en dehors du site : PutLoggingOptions
	en dehors du site : PutStorageConfiguration
	en dehors du site : UpdateAccessPolicy
	en dehors du site : UpdateAsset
	en dehors du site : UpdateAssetModel
	en dehors du site : UpdateAssetModelCompositeModel
	en dehors du site : UpdateAssetProperty
	en dehors du site : UpdateComputationModel
	en dehors du site : UpdateDashboard

Préfixe de service	Actions
	en dehors du site : UpdateDataset en dehors du site : UpdateGateway en dehors du site : UpdateGatewayCapabilityConfiguration en dehors du site : UpdatePortal en dehors du site : UpdateProject

Préfixe de service	Actions
iottwinmaker	iottwinmaker : CancelMetadataTransferJob iottwinmaker : CreateComponentType iottwinmaker : CreateEntity iottwinmaker : CreateMetadataTransferJob iottwinmaker : CreateScene iottwinmaker : CreateSyncJob iottwinmaker : CreateWorkspace iottwinmaker : DeleteComponentType iottwinmaker : DeleteEntity iottwinmaker : DeleteScene iottwinmaker : DeleteSyncJob iottwinmaker : DeleteWorkspace iottwinmaker : ExecuteQuery iottwinmaker : GetMetadataTransferJob iottwinmaker : GetPricingPlan iottwinmaker : GetScene iottwinmaker : GetSyncJob iottwinmaker : ListComponentTypes iottwinmaker : ListComponents iottwinmaker : ListEntities iottwinmaker : ListMetadataTransferJobs

Préfixe de service	Actions
	iottwinmaker : ListProperties iottwinmaker : ListScenes iottwinmaker : ListSyncJobs iottwinmaker : ListSyncResources iottwinmaker : ListWorkspaces iottwinmaker : UpdateComponentType iottwinmaker : UpdateEntity iottwinmaker : UpdatePricingPlan iottwinmaker : UpdateScene iottwinmaker : UpdateWorkspace

Préfixe de service	Actions
iotwireless	IoT sans fil : AssociateAwsAccountWithPartnerAccount
	IoT sans fil : AssociateMulticastGroupWithFuotaTask
	IoT sans fil : AssociateWirelessDeviceWithFuotaTask
	IoT sans fil : AssociateWirelessDeviceWithMulticastGroup
	IoT sans fil : AssociateWirelessDeviceWithThing
	IoT sans fil : AssociateWirelessGatewayWithCertificate
	IoT sans fil : AssociateWirelessGatewayWithThing
	IoT sans fil : CancelMulticastGroupSession
	IoT sans fil : CreateDestination
	IoT sans fil : CreateDeviceProfile
	IoT sans fil : CreateFuotaTask
	IoT sans fil : CreateMulticastGroup
	IoT sans fil : CreateNetworkAnalyzerConfiguration
	IoT sans fil : CreateServiceProfile
	IoT sans fil : CreateWirelessDevice
	IoT sans fil : CreateWirelessGateway
	IoT sans fil : CreateWirelessGatewayTask
	IoT sans fil : CreateWirelessGatewayTaskDefinition
	IoT sans fil : DeleteDestination
	IoT sans fil : DeleteDeviceProfile
	IoT sans fil : DeleteFuotaTask

Préfixe de service	Actions
	IoT sans fil : DeleteMulticastGroup
	IoT sans fil : DeleteNetworkAnalyzerConfiguration
	IoT sans fil : DeleteQueuedMessages
	IoT sans fil : DeleteServiceProfile
	IoT sans fil : DeleteWirelessDevice
	IoT sans fil : DeleteWirelessDeviceImportTask
	IoT sans fil : DeleteWirelessGateway
	IoT sans fil : DeleteWirelessGatewayTask
	IoT sans fil : DeleteWirelessGatewayTaskDefinition
	IoT sans fil : DeregisterWirelessDevice
	IoT sans fil : DisassociateAwsAccountFromPartnerAccount
	IoT sans fil : DisassociateMulticastGroupFromFuotaTask
	IoT sans fil : DisassociateWirelessDeviceFromFuotaTask
	IoT sans fil : DisassociateWirelessDeviceFromMulticastGroup
	IoT sans fil : DisassociateWirelessDeviceFromThing
	IoT sans fil : DisassociateWirelessGatewayFromCertificate
	IoT sans fil : DisassociateWirelessGatewayFromThing
	IoT sans fil : GetDestination
	IoT sans fil : GetDeviceProfile
	IoT sans fil : GetEventConfigurationByResourceTypes
	IoT sans fil : GetFuotaTask

Préfixe de service	Actions
	IoT sans fil : GetLogLevelsByResourceTypes
	IoT sans fil : GetMetricConfiguration
	IoT sans fil : GetMetrics
	IoT sans fil : GetMulticastGroup
	IoT sans fil : GetMulticastGroupSession
	IoT sans fil : GetNetworkAnalyzerConfiguration
	IoT sans fil : GetPartnerAccount
	IoT sans fil : GetPosition
	IoT sans fil : GetPositionConfiguration
	IoT sans fil : GetPositionEstimate
	IoT sans fil : GetResourceEventConfiguration
	IoT sans fil : GetResourceLogLevel
	IoT sans fil : GetResourcePosition
	IoT sans fil : GetServiceEndpoint
	IoT sans fil : GetServiceProfile
	IoT sans fil : GetWirelessDevice
	IoT sans fil : GetWirelessDeviceImportTask
	IoT sans fil : GetWirelessDeviceStatistics
	IoT sans fil : GetWirelessGateway
	IoT sans fil : GetWirelessGatewayCertificate
	IoT sans fil : GetWirelessGatewayFirmwareInformation

Préfixe de service	Actions
	IoT sans fil : GetWirelessGatewayStatistics
	IoT sans fil : GetWirelessGatewayTask
	IoT sans fil : GetWirelessGatewayTaskDefinition
	IoT sans fil : ListDestinations
	IoT sans fil : ListDeviceProfiles
	IoT sans fil : ListDevicesForWirelessDeviceImportTask
	IoT sans fil : ListEventConfigurations
	IoT sans fil : ListFuotaTasks
	IoT sans fil : ListMulticastGroups
	IoT sans fil : ListMulticastGroupsByFuotaTask
	IoT sans fil : ListNetworkAnalyzerConfigurations
	IoT sans fil : ListPartnerAccounts
	IoT sans fil : ListPositionConfigurations
	IoT sans fil : ListQueuedMessages
	IoT sans fil : ListServiceProfiles
	IoT sans fil : ListWirelessDeviceImportTasks
	IoT sans fil : ListWirelessDevices
	IoT sans fil : ListWirelessGatewayTaskDefinitions
	IoT sans fil : ListWirelessGateways
	IoT sans fil : PutPositionConfiguration
	IoT sans fil : PutResourceLogLevel

Préfixe de service	Actions
	IoT sans fil : ResetAllResourceLogLevels
	IoT sans fil : ResetResourceLogLevel
	IoT sans fil : SendDataToMulticastGroup
	IoT sans fil : SendDataToWirelessDevice
	IoT sans fil : StartBulkAssociateWirelessDeviceWithMulticastGroup
	IoT sans fil : StartBulkDisassociateWirelessDeviceFromMulticastGroup
	IoT sans fil : StartFuotaTask
	IoT sans fil : StartMulticastGroupSession
	IoT sans fil : StartNetworkAnalyzerStream
	IoT sans fil : StartSingleWirelessDeviceImportTask
	IoT sans fil : StartWirelessDeviceImportTask
	IoT sans fil : TestWirelessDevice
	IoT sans fil : UpdateDestination
	IoT sans fil : UpdateEventConfigurationByResourceTypes
	IoT sans fil : UpdateFuotaTask
	IoT sans fil : UpdateLogLevelsByResourceTypes
	IoT sans fil : UpdateMetricConfiguration
	IoT sans fil : UpdateMulticastGroup
	IoT sans fil : UpdateNetworkAnalyzerConfiguration
	IoT sans fil : UpdatePartnerAccount

Préfixe de service	Actions
	IoT sans fil : UpdatePosition
	IoT sans fil : UpdateResourceEventConfiguration
	IoT sans fil : UpdateResourcePosition
	IoT sans fil : UpdateWirelessDevice
	IoT sans fil : UpdateWirelessDeviceImportTask
	IoT sans fil : UpdateWirelessGateway

Préfixe de service	Actions
ivs	vis : BatchGetChannel vis : BatchGetStreamKey vis : BatchStartViewerSessionRevocation vis : CreateAdConfiguration vis : CreateChannel vis : CreateEncoderConfiguration vis : CreateIngestConfiguration vis : CreateParticipantToken vis : CreatePlaybackRestrictionPolicy vis : CreateRecordingConfiguration vis : CreateStorageConfiguration vis : CreateStreamKey vis : DeleteAdConfiguration vis : DeleteChannel vis : DeleteEncoderConfiguration vis : DeleteIngestConfiguration vis : DeletePlaybackKeyPair vis : DeletePlaybackRestrictionPolicy vis : DeletePublicKey vis : DeleteRecordingConfiguration vis : DeleteStorageConfiguration

Préfixe de service	Actions
	vis : DeleteStreamKey vis : DisconnectParticipant vis : GetAdConfiguration vis : GetChannel vis : GetComposition vis : GetEncoderConfiguration vis : GetIngestConfiguration vis : GetParticipant vis : GetPlaybackKeyPair vis : GetPlaybackRestrictionPolicy vis : GetPublicKey vis : GetRecordingConfiguration vis : GetStorageConfiguration vis : GetStream vis : GetStreamKey vis : GetStreamSession vis : ImportPlaybackKeyPair vis : ImportPublicKey vis : InsertAdBreak vis : ListAdConfigurations vis : ListChannels

Préfixe de service	Actions
	vis : ListCompositions vis : ListEncoderConfigurations vis : ListIngestConfigurations vis : ListParticipantEvents vis : ListParticipantReplicas vis : ListParticipants vis : ListPlaybackKeyPairs vis : ListPlaybackRestrictionPolicies vis : ListPublicKeys vis : ListRecordingConfigurations vis : ListStorageConfigurations vis : ListStreamKeys vis : ListStreamSessions vis : ListStreams vis : PutMetadata vis : StartComposition vis : StartViewerSessionRevocation vis : StopComposition vis : StopStream vis : UpdateAdConfiguration vis : UpdateChannel

Préfixe de service	Actions
	vis : UpdateIngestConfiguration vis : UpdatePlaybackRestrictionPolicy
ivschat	visa : CreateChatToken visa : CreateLoggingConfiguration visa : CreateRoom visa : DeleteLoggingConfiguration visa : DeleteMessage visa : DeleteRoom visa : DisconnectUser visa : GetLoggingConfiguration visa : GetRoom visa : ListLoggingConfigurations visa : ListRooms visa : SendEvent visa : UpdateLoggingConfiguration visa : UpdateRoom

Préfixe de service	Actions
kafka	Kafka : BatchAssociateScramSecret Kafka : BatchDisassociateScramSecret Kafka : CreateCluster Source : V2 CreateCluster Kafka : CreateConfiguration Kafka : CreateReplicator Kafka : CreateVpcConnection Kafka : DeleteCluster Kafka : DeleteClusterPolicy Kafka : DeleteConfiguration Kafka : DeleteReplicator Kafka : DeleteVpcConnection Kafka : DescribeCluster Kafka : DescribeClusterOperation Kafka : DescribeClusterOperationV2 Source : V2 DescribeCluster Kafka : DescribeConfiguration Kafka : DescribeConfigurationRevision Kafka : DescribeVpcConnection Kafka : GetBootstrapBrokers Kafka : GetClusterPolicy

Préfixe de service	Actions
	Kafka : GetCompatibleKafkaVersions Kafka : ListClientVpcConnections Kafka : ListClusterOperations Kafka : ListClusterOperationsV2 Kafka : ListClusters Source : V2 ListClusters Kafka : ListConfigurationRevisions Kafka : ListConfigurations Kafka : ListKafkaVersions Kafka : ListNodes Kafka : ListReplicators Kafka : ListScramSecrets Kafka : ListVpcConnections Kafka : PutClusterPolicy Kafka : RebootBroker Kafka : RejectClientVpcConnection Kafka : UpdateBrokerCount Kafka : UpdateBrokerStorage Kafka : UpdateBrokerType Kafka : UpdateClusterConfiguration Kafka : UpdateClusterKafkaVersion

Préfixe de service	Actions
	Kafka : UpdateConfiguration Kafka : UpdateConnectivity Kafka : UpdateMonitoring Kafka : UpdateRebalancing Kafka : UpdateReplicationInfo Kafka : UpdateSecurity Kafka : UpdateStorage
kafkaconnect	Kafka Connect : CreateConnector Kafka Connect : CreateCustomPlugin Kafka Connect : CreateWorkerConfiguration Kafka Connect : DeleteConnector Kafka Connect : DeleteCustomPlugin Kafka Connect : DeleteWorkerConfiguration Kafka Connect : DescribeConnector Kafka Connect : DescribeCustomPlugin Kafka Connect : DescribeWorkerConfiguration Kafka Connect : ListConnectorOperations Kafka Connect : ListConnectors Kafka Connect : ListCustomPlugins Kafka Connect : ListWorkerConfigurations Kafka Connect : UpdateConnector

Préfixe de service	Actions
kendra	kendra : AssociateEntitiesToExperience kendra : AssociatePersonasToEntities kendra : BatchDeleteDocument kendra : BatchDeleteFeaturedResultsSet kendra : BatchGetDocumentStatus kendra : BatchPutDocument kendra : ClearQuerySuggestions kendra : CreateAccessControlConfiguration kendra : CreateDataSource kendra : CreateExperience kendra : CreateFaq kendra : CreateFeaturedResultsSet kendra : CreateIndex kendra : CreateQuerySuggestionsBlockList kendra : CreateThesaurus kendra : DeleteDataSource kendra : DeleteExperience kendra : DeleteFaq kendra : DeleteIndex kendra : DeletePrincipalMapping kendra : DeleteQuerySuggestionsBlockList

Préfixe de service	Actions
	kendra : DeleteThesaurus kendra : DescribeAccessControlConfiguration kendra : DescribeDataSource kendra : DescribeExperience kendra : DescribeFaq kendra : DescribeFeaturedResultsSet kendra : DescribeIndex kendra : DescribePrincipalMapping kendra : DescribeQuerySuggestionsBlockList kendra : DescribeQuerySuggestionsConfig kendra : DescribeThesaurus kendra : DisassociateEntitiesFromExperience kendra : DisassociatePersonasFromEntities kendra : GetQuerySuggestions kendra : GetSnapshots kendra : ListAccessControlConfigurations kendra : ListDataSourceSyncJobs kendra : ListDataSources kendra : ListEntityPersonas kendra : ListExperienceEntities kendra : ListExperiences

Préfixe de service	Actions
	kendra : ListFaqs kendra : ListFeaturedResultsSets kendra : ListGroupsOlderThanOrderingId kendra : ListIndices kendra : ListQuerySuggestionsBlockLists kendra : ListThesauri kendra : PutPrincipalMapping kendra:Query kendra:Retrieve kendra : StartDataSourceSyncJob kendra : StopDataSourceSyncJob kendra : SubmitFeedback kendra : UpdateDataSource kendra : UpdateExperience kendra : UpdateFeaturedResultsSet kendra : UpdateIndex kendra : UpdateQuerySuggestionsBlockList kendra : UpdateQuerySuggestionsConfig kendra : UpdateThesaurus

Préfixe de service	Actions
kinesis	kinésie : CreateStream
	kinésie : DecreaseStreamRetentionPeriod
	kinésie : DeleteStream
	kinésie : DeregisterStreamConsumer
	kinésie : DescribeAccountSettings
	kinésie : DescribeLimits
	kinésie : DescribeStream
	kinésie : DescribeStreamConsumer
	kinésie : DescribeStreamSummary
	kinésie : DisableEnhancedMonitoring
	kinésie : EnableEnhancedMonitoring
	kinésie : IncreaseStreamRetentionPeriod
	kinésie : ListShards
	kinésie : ListStreamConsumers
	kinésie : ListStreams
	kinésie : MergeShards
	kinésie : RegisterStreamConsumer
	kinésie : SplitShard
	kinésie : StartStreamEncryption
	kinésie : StopStreamEncryption
	kinésie : UpdateAccountSettings

Préfixe de service	Actions
	kinésie : UpdateShardCount kinésie : UpdateStreamMode

Préfixe de service	Actions
kinesisanalytics	Kinesis Analytics : AddApplicationCloudWatchLoggingOption Kinesis Analytics : AddApplicationInput Kinesis Analytics : AddApplicationInputProcessingConfiguration Kinesis Analytics : AddApplicationOutput Kinesis Analytics : AddApplicationReferenceDataSource Kinesis Analytics : AddApplicationVpcConfiguration Kinesis Analytics : CreateApplication Kinesis Analytics : CreateApplicationPresignedUrl Kinesis Analytics : CreateApplicationSnapshot Kinesis Analytics : DeleteApplication Kinesis Analytics : DeleteApplicationCloudWatchLoggingOption Kinesis Analytics : DeleteApplicationInputProcessingConfiguration Kinesis Analytics : DeleteApplicationOutput Kinesis Analytics : DeleteApplicationReferenceDataSource Kinesis Analytics : DeleteApplicationSnapshot Kinesis Analytics : DeleteApplicationVpcConfiguration Kinesis Analytics : DescribeApplication Kinesis Analytics : DescribeApplicationOperation Kinesis Analytics : DescribeApplicationSnapshot Kinesis Analytics : DescribeApplicationVersion Kinesis Analytics : DiscoverInputSchema

Préfixe de service	Actions
	Kinesis Analytics : ListApplicationOperations Kinesis Analytics : ListApplicationSnapshots Kinesis Analytics : ListApplicationVersions Kinesis Analytics : ListApplications Kinesis Analytics : RollbackApplication Kinesis Analytics : StartApplication Kinesis Analytics : StopApplication Kinesis Analytics : UpdateApplication Kinesis Analytics : UpdateApplicationMaintenanceConfiguration

Préfixe de service	Actions
kms	km : CancelKeyDeletion
	km : ConnectCustomKeyStore
	km : CreateAlias
	km : CreateCustomKeyStore
	km : CreateGrant
	km : CreateKey
	kms:Decrypt
	km : DeleteAlias
	km : DeleteCustomKeyStore
	km : DeleteImportedKeyMaterial
	km : DeriveSharedSecret
	km : DescribeCustomKeyStores
	km : DescribeKey
	km : DisableKey
	km : DisableKeyRotation
	km : DisconnectCustomKeyStore
	km : EnableKey
	km : EnableKeyRotation
	kms:Encrypt
	km : GenerateDataKey
	km : GenerateDataKeyPair

Préfixe de service	Actions
	km : GenerateDataKeyPairWithoutPlaintext
	km : GenerateDataKeyWithoutPlaintext
	km : GenerateMac
	km : GenerateRandom
	km : GetKeyLastUsage
	km : GetKeyPolicy
	km : GetKeyRotationStatus
	km : GetParametersForImport
	km : GetPublicKey
	km : ImportKeyMaterial
	km : ListAliases
	km : ListGrants
	km : ListKeyPolicies
	km : ListKeyRotations
	km : ListKeys
	km : ListRetirableGrants
	km : ReplicateKey
	km : RetireGrant
	km : RevokeGrant
	km : RotateKeyOnDemand
	km : ScheduleKeyDeletion

Préfixe de service	Actions
	kms:Sign km : UpdateAlias km : UpdateCustomKeyStore km : UpdateKeyDescription km : UpdatePrimaryRegion kms:Verify km : VerifyMac

Préfixe de service	Actions
lambda	lambda : AddLayerVersionPermission lambda : AddPermission lambda : CreateAlias lambda : CreateCodeSigningConfig lambda : CreateEventSourceMapping lambda : CreateFunction lambda : CreateFunctionUrlConfig lambda : DeleteAlias lambda : DeleteCapacityProvider lambda : DeleteCodeSigningConfig lambda : DeleteEventSourceMapping lambda : DeleteFunction lambda : DeleteFunctionCodeSigningConfig lambda : DeleteFunctionConcurrency lambda : DeleteFunctionEventInvokeConfig lambda : DeleteFunctionUrlConfig lambda : DeleteLayerVersion lambda : DeleteProvisionedConcurrencyConfig lambda : GetAccountSettings lambda : GetAlias lambda : GetCapacityProvider

Préfixe de service	Actions
	lambda : GetCodeSigningConfig lambda : GetEventSourceMapping lambda : GetFunction lambda : GetFunctionCodeSigningConfig lambda : GetFunctionConcurrency lambda : GetFunctionConfiguration lambda : GetFunctionEventInvokeConfig lambda : GetFunctionRecursionConfig lambda : GetFunctionScalingConfig lambda : GetFunctionUrlConfig lambda : GetLayerVersion lambda : GetLayerVersionPolicy lambda : GetPolicy lambda : GetProvisionedConcurrencyConfig lambda : GetRuntimeManagementConfig lambda : ListAliases lambda : ListCapacityProviders lambda : ListCodeSigningConfigs lambda : ListDurableExecutionsByFunction lambda : ListEventSourceMappings lambda : ListFunctionEventInvokeConfigs

Préfixe de service	Actions
	lambda : ListFunctionUrlConfigs lambda : ListFunctions lambda : ListFunctionsByCodeSigningConfig lambda : ListLayerVersions lambda : ListLayers lambda : ListProvisionedConcurrencyConfigs lambda : ListVersionsByFunction lambda : PublishLayerVersion lambda : PublishVersion lambda : PutFunctionCodeSigningConfig lambda : PutFunctionConcurrency lambda : PutFunctionEventInvokeConfig lambda : PutFunctionRecursionConfig lambda : PutFunctionScalingConfig lambda : PutProvisionedConcurrencyConfig lambda : PutRuntimeManagementConfig lambda : RemoveLayerVersionPermission lambda : RemovePermission lambda : UpdateAlias lambda : UpdateCapacityProvider lambda : UpdateCodeSigningConfig

Préfixe de service	Actions
	lambda : UpdateEventSourceMapping lambda : UpdateFunctionCode lambda : UpdateFunctionConfiguration lambda : UpdateFunctionEventInvokeConfig lambda : UpdateFunctionUrlConfig

Préfixe de service	Actions
lex	Alex : BatchCreateCustomVocabularyItem
	Alex : BatchDeleteCustomVocabularyItem
	Alex : BatchUpdateCustomVocabularyItem
	Alex : BuildBotLocale
	Alex : CreateBotAlias
	Alex : CreateBotReplica
	Alex : CreateBotVersion
	Alex : CreateExport
	Alex : CreateIntentVersion
	Alex : CreateResourcePolicy
	Alex : CreateSlotTypeVersion
	Alex : CreateTestSetDiscrepancyReport
	Alex : CreateUploadUrl
	Alex : DeleteBot
	Alex : DeleteBotAnalyzerRecommendation
	Alex : DeleteBotChannelAssociation
	Alex : DeleteBotReplica
	Alex : DeleteExport
	Alex : DeleteImport
	Alex : DeleteIntentVersion
	Alex : DeleteResourcePolicy

Préfixe de service	Actions
	Alex : DeleteSlotTypeVersion
	Alex : DeleteTestSet
	Alex : DeleteUtterances
	Alex : DescribeBotAlias
	Alex : DescribeBotAnalyzerRecommendation
	Alex : DescribeBotRecommendation
	Alex : DescribeBotReplica
	Alex : DescribeBotResourceGeneration
	Alex : DescribeBotVersion
	Alex : DescribeCustomVocabularyMetadata
	Alex : DescribeExport
	Alex : DescribeImport
	Alex : DescribeResourcePolicy
	Alex : DescribeTestExecution
	Alex : DescribeTestSet
	Alex : DescribeTestSetDiscrepancyReport
	Alex : DescribeTestSetGeneration
	Alex : GenerateBotElement
	Alex : GetBot
	Alex : GetBotAlias
	Alex : GetBotAliases

Préfixe de service	Actions
	Alex : GetBotChannelAssociation
	Alex : GetBotChannelAssociations
	Alex : GetBotVersions
	Alex : GetBots
	Alex : GetBuiltinIntent
	Alex : GetBuiltinIntents
	Alex : GetBuiltinSlotTypes
	Alex : GetExport
	Alex : GetImport
	Alex : GetIntent
	Alex : GetIntentVersions
	Alex : GetIntents
	Alex : GetMigration
	Alex : GetMigrations
	Alex : GetSlotType
	Alex : GetSlotTypeVersions
	Alex : GetSlotTypes
	Alex : GetTestExecutionArtifactsUrl
	Alex : GetUtterancesView
	Alex : ListBotAliasReplicas
	Alex : ListBotAliases

Préfixe de service	Actions
	Alex : ListBotRecommendations
	Alex : ListBotReplicas
	Alex : ListBotResourceGenerations
	Alex : ListBotVersionReplicas
	Alex : ListBotVersions
	Alex : ListBots
	Alex : ListBuiltInIntents
	Alex : ListBuiltInSlotTypes
	Alex : ListCustomVocabularyItems
	Alex : ListExports
	Alex : ListImports
	Alex : ListIntentMetrics
	Alex : ListIntentPaths
	Alex : ListRecommendedIntents
	Alex : ListSessionAnalyticsData
	Alex : ListSessionMetrics
	Alex : ListTestExecutionResultItems
	Alex : ListTestExecutions
	Alex : ListTestSets
	Alex : PutBot
	Alex : PutBotAlias

Préfixe de service	Actions
	Alex : PutIntent
	Alex : PutSlotType
	Alex : SearchAssociatedTranscripts
	Alex : StartBotAnalyzer
	Alex : StartBotRecommendation
	Alex : StartImport
	Alex : StartMigration
	Alex : StartTestExecution
	Alex : StartTestSetGeneration
	Alex : StopBotAnalyzer
	Alex : StopBotRecommendation
	Alex : UpdateBotAlias
	Alex : UpdateBotRecommendation
	Alex : UpdateExport
	Alex : UpdateResourcePolicy

Préfixe de service	Actions
license-manager-linux-subscriptions	License-Manager-Linux-Subscriptions : DeregisterSubscriptionProvider License-Manager-Linux-Subscriptions : GetRegisteredSubscriptionProvider License-Manager-Linux-Subscriptions : GetServiceSettings License-Manager-Linux-Subscriptions : ListLinuxSubscriptionInstances License-Manager-Linux-Subscriptions : ListLinuxSubscriptions License-Manager-Linux-Subscriptions : ListRegisteredSubscriptionProviders License-Manager-Linux-Subscriptions : RegisterSubscriptionProvider License-Manager-Linux-Subscriptions : UpdateServiceSettings

Préfixe de service	Actions
lightsail	voile lumineuse : AllocateStaticIp
	voile lumineuse : AttachCertificateToDistribution
	voile lumineuse : AttachDisk
	voile lumineuse : AttachInstancesToLoadBalancer
	voile lumineuse : AttachLoadBalancerTlsCertificate
	voile lumineuse : AttachStaticIp
	voile lumineuse : CloseInstancePublicPorts
	voile lumineuse : CopySnapshot
	voile lumineuse : CreateBucket
	voile lumineuse : CreateBucketAccessKey
	voile lumineuse : CreateCertificate
	voile lumineuse : CreateCloudFormationStack
	voile lumineuse : CreateContactMethod
	voile lumineuse : CreateContainerService
	voile lumineuse : CreateContainerServiceDeployment
	voile lumineuse : CreateContainerServiceRegistryLogin
	voile lumineuse : CreateDisk
	voile lumineuse : CreateDiskFromSnapshot
	voile lumineuse : CreateDiskSnapshot
	voile lumineuse : CreateDistribution
	voile lumineuse : CreateDomain

Préfixe de service	Actions
	voile lumineuse : CreateGUISessionAccessDetails
	voile lumineuse : CreateInstanceSnapshot
	voile lumineuse : CreateInstances
	voile lumineuse : CreateInstancesFromSnapshot
	voile lumineuse : CreateKeyPair
	voile lumineuse : CreateLoadBalancer
	voile lumineuse : CreateLoadBalancerTlsCertificate
	voile lumineuse : CreateRelationalDatabase
	voile lumineuse : CreateRelationalDatabaseFromSnapshot
	voile lumineuse : CreateRelationalDatabaseSnapshot
	voile lumineuse : DeleteAlarm
	voile lumineuse : DeleteAutoSnapshot
	voile lumineuse : DeleteBucket
	voile lumineuse : DeleteBucketAccessKey
	voile lumineuse : DeleteCertificate
	voile lumineuse : DeleteContactMethod
	voile lumineuse : DeleteContainerImage
	voile lumineuse : DeleteContainerService
	voile lumineuse : DeleteDisk
	voile lumineuse : DeleteDiskSnapshot
	voile lumineuse : DeleteDistribution

Préfixe de service	Actions
	voile lumineuse : DeleteDomain
	voile lumineuse : DeleteDomainEntry
	voile lumineuse : DeleteInstance
	voile lumineuse : DeleteInstanceSnapshot
	voile lumineuse : DeleteKeyPair
	voile lumineuse : DeleteKnownHostKeys
	voile lumineuse : DeleteLoadBalancer
	voile lumineuse : DeleteLoadBalancerTlsCertificate
	voile lumineuse : DeleteRelationalDatabase
	voile lumineuse : DeleteRelationalDatabaseSnapshot
	voile lumineuse : DetachCertificateFromDistribution
	voile lumineuse : DetachDisk
	voile lumineuse : DetachInstancesFromLoadBalancer
	voile lumineuse : DetachStaticIp
	voile lumineuse : DisableAddOn
	voile lumineuse : DownloadDefaultKeyPair
	voile lumineuse : EnableAddOn
	voile lumineuse : ExportSnapshot
	voile lumineuse : GetActiveNames
	voile lumineuse : GetAlarms
	voile lumineuse : GetAutoSnapshots

Préfixe de service	Actions
	voile lumineuse : GetBlueprints
	voile lumineuse : GetBucketAccessKeys
	voile lumineuse : GetBucketBundles
	voile lumineuse : GetBucketMetricData
	voile lumineuse : GetBuckets
	voile lumineuse : GetBundles
	voile lumineuse : GetCertificates
	voile lumineuse : GetCloudFormationStackRecords
	voile lumineuse : GetContactMethods
	voile lumineuse : GetContainerAPIMetadata
	voile lumineuse : GetContainerImages
	voile lumineuse : GetContainerLog
	voile lumineuse : GetContainerServiceDeployments
	voile lumineuse : GetContainerServiceMetricData
	voile lumineuse : GetContainerServicePowers
	voile lumineuse : GetContainerServices
	voile lumineuse : GetCostEstimate
	voile lumineuse : GetDisk
	voile lumineuse : GetDiskSnapshot
	voile lumineuse : GetDiskSnapshots
	voile lumineuse : GetDisks

Préfixe de service	Actions
	voile lumineuse : GetDistributionBundles
	voile lumineuse : GetDistributionLatestCacheReset
	voile lumineuse : GetDistributionMetricData
	voile lumineuse : GetDistributions
	voile lumineuse : GetDomain
	voile lumineuse : GetExportSnapshotRecords
	voile lumineuse : GetInstance
	voile lumineuse : GetInstanceMetricData
	voile lumineuse : GetInstancePortStates
	voile lumineuse : GetInstanceSnapshot
	voile lumineuse : GetInstanceSnapshots
	voile lumineuse : GetInstanceState
	voile lumineuse : GetInstances
	voile lumineuse : GetKeyPair
	voile lumineuse : GetKeyPairs
	voile lumineuse : GetLoadBalancer
	voile lumineuse : GetLoadBalancerMetricData
	voile lumineuse : GetLoadBalancerTlsCertificates
	voile lumineuse : GetLoadBalancerTlsPolicies
	voile lumineuse : GetLoadBalancers
	voile lumineuse : GetOperation

Préfixe de service	Actions
	voile lumineuse : GetOperations
	voile lumineuse : GetOperationsForResource
	voile lumineuse : GetRegions
	voile lumineuse : GetRelationalDatabase
	voile lumineuse : GetRelationalDatabaseBlueprints
	voile lumineuse : GetRelationalDatabaseBundles
	voile lumineuse : GetRelationalDatabaseEvents
	voile lumineuse : GetRelationalDatabaseLogEvents
	voile lumineuse : GetRelationalDatabaseLogStreams
	voile lumineuse : GetRelationalDatabaseMasterUserPassword
	voile lumineuse : GetRelationalDatabaseMetricData
	voile lumineuse : GetRelationalDatabaseParameters
	voile lumineuse : GetRelationalDatabaseSnapshot
	voile lumineuse : GetRelationalDatabaseSnapshots
	voile lumineuse : GetRelationalDatabases
	voile lumineuse : GetSetupHistory
	voile lumineuse : GetStaticIp
	voile lumineuse : GetStaticIps
	voile lumineuse : ImportKeyPair
	voile lumineuse : IsVpcPeered
	voile lumineuse : OpenInstancePublicPorts

Préfixe de service	Actions
	voile lumineuse : PeerVpc
	voile lumineuse : PutAlarm
	voile lumineuse : PutInstancePublicPorts
	voile lumineuse : RebootInstance
	voile lumineuse : RebootRelationalDatabase
	voile lumineuse : RegisterContainerImage
	voile lumineuse : ReleaseStaticIp
	voile lumineuse : ResetDistributionCache
	voile lumineuse : SendContactMethodVerification
	voile lumineuse : SetIpAddressType
	voile lumineuse : SetResourceAccessForBucket
	voile lumineuse : SetupInstanceHttps
	lightsail:StartGUISession
	voile lumineuse : StartInstance
	voile lumineuse : StartRelationalDatabase
	lightsail:StopGUISession
	voile lumineuse : StopInstance
	voile lumineuse : StopRelationalDatabase
	voile lumineuse : TestAlarm
	voile lumineuse : UnpeerVpc
	voile lumineuse : UpdateBucket

Préfixe de service	Actions
	voile lumineuse : UpdateBucketBundle
	voile lumineuse : UpdateContainerService
	voile lumineuse : UpdateDistribution
	voile lumineuse : UpdateDistributionBundle
	voile lumineuse : UpdateDomainEntry
	voile lumineuse : UpdateInstanceMetadataOptions
	voile lumineuse : UpdateLoadBalancerAttribute
	voile lumineuse : UpdateRelationalDatabase
	voile lumineuse : UpdateRelationalDatabaseParameters

Préfixe de service	Actions
journaux	journaux : AssociateKmsKey journaux : AssociateSourceToS3TableIntegration journaux : CancelExportTask journaux : CancelImportTask journaux : CreateDelivery journaux : CreateExportTask journaux : CreateLogAnomalyDetector journaux : CreateLogGroup journaux : CreateLogStream journaux : CreateLookupTable journaux : DeleteDataProtectionPolicy journaux : DeleteDelivery journaux : DeleteDeliveryDestination journaux : DeleteDeliveryDestinationPolicy journaux : DeleteDeliverySource journaux : DeleteDestination journaux : DeleteIndexPolicy journaux : DeleteIntegration journaux : DeleteLogAnomalyDetector journaux : DeleteLogGroup journaux : DeleteLogStream

Préfixe de service	Actions
	journaux : DeleteLookupTable journaux : DeleteMetricFilter journaux : DeleteQueryDefinition journaux : DeleteResourcePolicy journaux : DeleteRetentionPolicy journaux : DeleteScheduledQuery journaux : DeleteSubscriptionFilter journaux : DeleteSyslogConfiguration journaux : DeleteTransformer journaux : DescribeAccountPolicies journaux : DescribeConfigurationTemplates journaux : DescribeDeliveries journaux : DescribeDeliveryDestinations journaux : DescribeDeliverySources journaux : DescribeDestinations journaux : DescribeExportTasks journaux : DescribeFieldIndexes journaux : DescribeImportTaskBatches journaux : DescribeImportTasks journaux : DescribeIndexPolicies journaux : DescribeLogGroups

Préfixe de service	Actions
	journaux : DescribeLogStreams journaux : DescribeLookupTables journaux : DescribeMetricFilters journaux : DescribeQueries journaux : DescribeQueryDefinitions journaux : DescribeResourcePolicies journaux : DescribeSubscriptionFilters journaux : DisassociateKmsKey journaux : DisassociateSourceFromS3TableIntegration journaux : GetDataProtectionPolicy journaux : GetDelivery journaux : GetDeliveryDestination journaux : GetDeliveryDestinationPolicy journaux : GetDeliverySource journaux : GetIntegration journaux : GetLogAnomalyDetector journaux : GetLogFields journaux : GetLogGroupFields journaux : GetLogRecord journaux : GetLookupTable journaux : GetQueryResults

Préfixe de service	Actions
	journaux : GetScheduledQuery journaux : GetScheduledQueryHistory journaux : GetStorageTierPolicy journaux : GetTransformer journaux : ListAnomalies journaux : ListIntegrations journaux : ListLogAnomalyDetectors journaux : ListLogGroupsForQuery journaux : ListScheduledQueries journaux : ListSourcesForS3TableIntegration journaux : ListSyslogConfigurations journaux : PutBearerTokenAuthentication journaux : PutDataProtectionPolicy journaux : PutDeliveryDestination journaux : PutDeliveryDestinationPolicy journaux : PutDeliverySource journaux : PutDestination journaux : PutDestinationPolicy journaux : PutIndexPolicy journaux : PutIntegration journaux : PutLogGroupDeletionProtection

Préfixe de service	Actions
	journaux : PutMetricFilter journaux : PutQueryDefinition journaux : PutResourcePolicy journaux : PutRetentionPolicy journaux : PutStorageTierPolicy journaux : PutSubscriptionFilter journaux : PutSyslogConfiguration journaux : PutTransformer journaux : StartLiveTail journaux : StartQuery journaux : StopQuery journaux : TestMetricFilter journaux : TestTransformer journaux : UpdateAnomaly journaux : UpdateDeliveryConfiguration journaux : UpdateLogAnomalyDetector journaux : UpdateLookupTable

Préfixe de service	Actions
lookoutequipment	équipement de surveillance : CreateDataset
	équipement de surveillance : CreateInferenceScheduler
	équipement de surveillance : CreateLabel
	équipement de surveillance : CreateLabelGroup
	équipement de surveillance : CreateModel
	équipement de surveillance : DeleteDataset
	équipement de surveillance : DeleteInferenceScheduler
	équipement de surveillance : DeleteLabel
	équipement de surveillance : DeleteLabelGroup
	équipement de surveillance : DeleteModel
	équipement de surveillance : DeleteResourcePolicy
	équipement de surveillance : DeleteRetrainingScheduler
	équipement de surveillance : DescribeDataIngestionJob
	équipement de surveillance : DescribeDataset
	équipement de surveillance : DescribeInferenceScheduler
	équipement de surveillance : DescribeLabelGroup
	équipement de surveillance : DescribeModel
	équipement de surveillance : DescribeModelVersion
	équipement de surveillance : DescribeResourcePolicy
	équipement de surveillance : DescribeRetrainingScheduler
	lookoutequipment:Describelabel

Préfixe de service	Actions
	équipement de surveillance : ImportDataset
	équipement de surveillance : ImportModelVersion
	équipement de surveillance : ListDataIngestionJobs
	équipement de surveillance : ListDatasets
	équipement de surveillance : ListInferenceEvents
	équipement de surveillance : ListInferenceExecutions
	équipement de surveillance : ListInferenceSchedulers
	équipement de surveillance : ListLabelGroups
	équipement de surveillance : ListLabels
	équipement de surveillance : ListModelVersions
	équipement de surveillance : ListModels
	équipement de surveillance : ListRetrainingSchedulers
	équipement de surveillance : ListSensorStatistics
	équipement de surveillance : PutResourcePolicy
	équipement de surveillance : StartDataIngestionJob
	équipement de surveillance : StartInferenceScheduler
	équipement de surveillance : StartRetrainingScheduler
	équipement de surveillance : StopInferenceScheduler
	équipement de surveillance : StopRetrainingScheduler
	équipement de surveillance : UpdateActiveModelVersion
	équipement de surveillance : UpdateInferenceScheduler

Préfixe de service	Actions
	équipement de surveillance : UpdateLabelGroup équipement de surveillance : UpdateModel équipement de surveillance : UpdateRetrainingScheduler

Préfixe de service	Actions
lookoutmetrics	Métriques de surveillance : ActivateAnomalyDetector Métriques de surveillance : BackTestAnomalyDetector Métriques de surveillance : CreateAlert Métriques de surveillance : CreateAnomalyDetector Métriques de surveillance : CreateMetricSet Métriques de surveillance : DeactivateAnomalyDetector Métriques de surveillance : DeleteAlert Métriques de surveillance : DeleteAnomalyDetector Métriques de surveillance : DescribeAlert Métriques de surveillance : DescribeAnomalyDetectionExecutions Métriques de surveillance : DescribeAnomalyDetector Métriques de surveillance : DescribeMetricSet Métriques de surveillance : DetectMetricSetConfig Métriques de surveillance : GetAnomalyGroup Métriques de surveillance : GetDataQualityMetrics Métriques de surveillance : GetFeedback Métriques de surveillance : GetSampleData Métriques de surveillance : ListAlerts Métriques de surveillance : ListAnomalyDetectors Métriques de surveillance : ListAnomalyGroupRelatedMetrics Métriques de surveillance : ListAnomalyGroupSummaries

Préfixe de service	Actions
	Métriques de surveillance : ListAnomalyGroupTimeSeries Métriques de surveillance : ListMetricSets Métriques de surveillance : PutFeedback Métriques de surveillance : UpdateAlert Métriques de surveillance : UpdateAnomalyDetector Métriques de surveillance : UpdateMetricSet

Préfixe de service	Actions
m2	m2 : CancelBatchJobExecution
	m2 : CreateApplication
	m2 : CreateDataSetExportTask
	m2 : CreateDataSetImportTask
	m2 : CreateDeployment
	m2 : CreateEnvironment
	m2 : DeleteApplication
	m2 : DeleteApplicationFromEnvironment
	m2 : DeleteEnvironment
	m2 : GetApplication
	m2 : GetApplicationVersion
	m2 : GetBatchJobExecution
	m2 : GetDataSetDetails
	m2 : GetDataSetExportTask
	m2 : GetDataSetImportTask
	m2 : GetDeployment
	m2 : GetEnvironment
	m2 : GetSignedBluinsightsUrl
	m2 : ListApplicationVersions
	m2 : ListApplications
	m2 : ListBatchJobDefinitions

Préfixe de service	Actions
	m2 : ListBatchJobExecutions
	m2 : ListBatchJobRestartPoints
	m2 : ListDataSetExportHistory
	m2 : ListDataSetImportHistory
	m2 : ListDataSets
	m2 : ListDeployments
	m2 : ListEngineVersions
	m2 : ListEnvironments
	m2 : StartApplication
	m2 : StartBatchJob
	m2 : StopApplication
	m2 : UpdateApplication
	m2 : UpdateEnvironment

Préfixe de service	Actions
managedblockchain	blockchain gérée : CreateAccessor blockchain gérée : CreateMember blockchain gérée : CreateNetwork blockchain gérée : CreateNode blockchain gérée : CreateProposal blockchain gérée : DeleteAccessor blockchain gérée : DeleteMember blockchain gérée : DeleteNode blockchain gérée : GetAccessor blockchain gérée : GetMember blockchain gérée : GetNetwork blockchain gérée : GetNode blockchain gérée : GetProposal blockchain gérée : InvokeRpcPolygonMainnet blockchain gérée : InvokeRpcPolygonMumbaiTestnet blockchain gérée : ListAccessors blockchain gérée : ListInvitations blockchain gérée : ListMembers blockchain gérée : ListNetworks blockchain gérée : ListNodes blockchain gérée : ListProposalVotes

Préfixe de service	Actions
	blockchain gérée : ListProposals blockchain gérée : RejectInvitation blockchain gérée : UpdateMember blockchain gérée : UpdateNode blockchain gérée : VoteOnProposal

Préfixe de service	Actions
mediaconnect	connexion multimédia : AddBridgeOutputs
	connexion multimédia : AddBridgeSources
	connexion multimédia : AddFlowMediaStreams
	connexion multimédia : AddFlowOutputs
	connexion multimédia : AddFlowSources
	connexion multimédia : AddFlowVpcInterfaces
	connexion multimédia : CreateBridge
	connexion multimédia : CreateFlow
	connexion multimédia : CreateGateway
	connexion multimédia : DeleteBridge
	connexion multimédia : DeleteFlow
	connexion multimédia : DeleteGateway
	connexion multimédia : DeleteRouterInput
	connexion multimédia : DeleteRouterNetworkInterface
	connexion multimédia : DeleteRouterOutput
	connexion multimédia : DeregisterGatewayInstance
	connexion multimédia : DescribeBridge
	connexion multimédia : DescribeFlow
	connexion multimédia : DescribeFlowSourceMetadata
	connexion multimédia : DescribeFlowSourceThumbnail
	connexion multimédia : DescribeGateway

Préfixe de service	Actions
	connexion multimédia : DescribeGatewayInstance
	connexion multimédia : DescribeOffering
	connexion multimédia : DescribeReservation
	connexion multimédia : GetRouterInput
	connexion multimédia : GetRouterInputSourceMetadata
	connexion multimédia : GetRouterInputThumbnail
	connexion multimédia : GetRouterNetworkInterface
	connexion multimédia : GetRouterOutput
	connexion multimédia : GrantFlowEntitlements
	connexion multimédia : ListBridges
	connexion multimédia : ListEntitlements
	connexion multimédia : ListFlows
	connexion multimédia : ListGatewayInstances
	connexion multimédia : ListGateways
	connexion multimédia : ListOfferings
	connexion multimédia : ListReservations
	connexion multimédia : ListRouterInputs
	connexion multimédia : ListRouterNetworkInterfaces
	connexion multimédia : ListRouterOutputs
	connexion multimédia : PurchaseOffering
	connexion multimédia : RemoveBridgeOutput

Préfixe de service	Actions
	connexion multimédia : RemoveBridgeSource
	connexion multimédia : RemoveFlowMediaStream
	connexion multimédia : RemoveFlowOutput
	connexion multimédia : RemoveFlowSource
	connexion multimédia : RemoveFlowVpcInterface
	connexion multimédia : RestartRouterInput
	connexion multimédia : RestartRouterOutput
	connexion multimédia : RevokeFlowEntitlement
	connexion multimédia : StartFlow
	connexion multimédia : StartRouterInput
	connexion multimédia : StartRouterOutput
	connexion multimédia : StopFlow
	connexion multimédia : StopRouterInput
	connexion multimédia : StopRouterOutput
	connexion multimédia : TakeRouterInput
	connexion multimédia : UpdateBridge
	connexion multimédia : UpdateBridgeOutput
	connexion multimédia : UpdateBridgeSource
	connexion multimédia : UpdateBridgeState
	connexion multimédia : UpdateFlow
	connexion multimédia : UpdateFlowEntitlement

Préfixe de service	Actions
	connexion multimédia : UpdateFlowMediaStream connexion multimédia : UpdateGatewayInstance

Préfixe de service	Actions
mediaconvert	conversion de média : AssociateCertificate conversion de média : CancelJob conversion de média : CreateJob conversion de média : CreateJobTemplate conversion de média : CreatePreset conversion de média : CreateQueue conversion de média : CreateResourceShare conversion de média : DeleteJobTemplate conversion de média : DeletePolicy conversion de média : DeletePreset conversion de média : DeleteQueue conversion de média : DescribeEndpoints conversion de média : DisassociateCertificate conversion de média : GetJob conversion de média : GetJobTemplate conversion de média : GetPolicy conversion de média : GetPreset conversion de média : GetQueue conversion de média : ListJobTemplates conversion de média : ListJobs conversion de média : ListPresets

Préfixe de service	Actions
	conversion de média : ListQueues conversion de média : ListVersions mediaconvert:Probe conversion de média : PutPolicy conversion de média : SearchJobs conversion de média : UpdateJobTemplate conversion de média : UpdatePreset conversion de média : UpdateQueue

Préfixe de service	Actions
medialive	MediaLive : AcceptInputDeviceTransfer
	MediaLive : BatchDelete
	MediaLive : BatchStart
	MediaLive : BatchStop
	MediaLive : BatchUpdateSchedule
	MediaLive : CancelInputDeviceTransfer
	MediaLive : ClaimDevice
	MediaLive : CreateChannel
	MediaLive : CreateChannelPlacementGroup
	MediaLive : CreateCloudWatchAlarmTemplate
	MediaLive : CreateCloudWatchAlarmTemplateGroup
	MediaLive : CreateCluster
	MediaLive : CreateEventBridgeRuleTemplate
	MediaLive : CreateEventBridgeRuleTemplateGroup
	MediaLive : CreateInput
	MediaLive : CreateInputSecurityGroup
	MediaLive : CreateMultiplex
	MediaLive : CreateMultiplexProgram
	MediaLive : CreateNetwork
	MediaLive : CreateNode
	MediaLive : CreateNodeRegistrationScript

Préfixe de service	Actions
	MediaLive : CreatePartnerInput
	MediaLive : CreateSdiSource
	MediaLive : CreateSignalMap
	MediaLive : DeleteChannel
	MediaLive : DeleteChannelPlacementGroup
	MediaLive : DeleteCloudWatchAlarmTemplate
	MediaLive : DeleteCloudWatchAlarmTemplateGroup
	MediaLive : DeleteCluster
	MediaLive : DeleteEventBridgeRuleTemplate
	MediaLive : DeleteEventBridgeRuleTemplateGroup
	MediaLive : DeleteInput
	MediaLive : DeleteInputSecurityGroup
	MediaLive : DeleteMultiplex
	MediaLive : DeleteMultiplexProgram
	MediaLive : DeleteNetwork
	MediaLive : DeleteNode
	MediaLive : DeleteReservation
	MediaLive : DeleteSchedule
	MediaLive : DeleteSdiSource
	MediaLive : DeleteSignalMap
	MediaLive : DescribeAccountConfiguration

Préfixe de service	Actions
	MediaLive : DescribeChannel
	MediaLive : DescribeChannelPlacementGroup
	MediaLive : DescribeCluster
	MediaLive : DescribeInput
	MediaLive : DescribeInputDevice
	MediaLive : DescribeInputDeviceThumbnail
	MediaLive : DescribeInputSecurityGroup
	MediaLive : DescribeMultiplex
	MediaLive : DescribeMultiplexProgram
	MediaLive : DescribeNetwork
	MediaLive : DescribeNode
	MediaLive : DescribeOffering
	MediaLive : DescribeReservation
	MediaLive : DescribeSchedule
	MediaLive : DescribeSdiSource
	MediaLive : DescribeThumbnails
	MediaLive : GetCloudWatchAlarmTemplate
	MediaLive : GetCloudWatchAlarmTemplateGroup
	MediaLive : GetEventBridgeRuleTemplate
	MediaLive : GetEventBridgeRuleTemplateGroup
	MediaLive : GetSignalMap

Préfixe de service	Actions
	MediaLive : ListAlerts
	MediaLive : ListChannelPlacementGroups
	MediaLive : ListChannels
	MediaLive : ListCloudWatchAlarmTemplateGroups
	MediaLive : ListCloudWatchAlarmTemplates
	MediaLive : ListClusterAlerts
	MediaLive : ListClusters
	MediaLive : ListEventBridgeRuleTemplateGroups
	MediaLive : ListEventBridgeRuleTemplates
	MediaLive : ListInputDeviceTransfers
	MediaLive : ListInputDevices
	MediaLive : ListInputSecurityGroups
	MediaLive : ListInputs
	MediaLive : ListMultiplexAlerts
	MediaLive : ListMultiplexPrograms
	MediaLive : ListMultiplexes
	MediaLive : ListNetworks
	MediaLive : ListNodes
	MediaLive : ListOfferings
	MediaLive : ListReservations
	MediaLive : ListSdiSources

Préfixe de service	Actions
	MediaLive : ListSignalMaps
	MediaLive : ListVersions
	MediaLive : PurchaseOffering
	MediaLive : RebootInputDevice
	MediaLive : RejectInputDeviceTransfer
	MediaLive : RestartChannelPipelines
	MediaLive : StartChannel
	MediaLive : StartDeleteMonitorDeployment
	MediaLive : StartInputDevice
	MediaLive : StartInputDeviceMaintenanceWindow
	MediaLive : StartMonitorDeployment
	MediaLive : StartMultiplex
	MediaLive : StartUpdateSignalMap
	MediaLive : StopChannel
	MediaLive : StopInputDevice
	MediaLive : StopMultiplex
	MediaLive : TransferInputDevice
	MediaLive : UpdateAccountConfiguration
	MediaLive : UpdateChannel
	MediaLive : UpdateChannelClass
	MediaLive : UpdateChannelPlacementGroup

Préfixe de service	Actions
	MediaLive : UpdateCloudWatchAlarmTemplate
	MediaLive : UpdateCloudWatchAlarmTemplateGroup
	MediaLive : UpdateCluster
	MediaLive : UpdateEventBridgeRuleTemplate
	MediaLive : UpdateEventBridgeRuleTemplateGroup
	MediaLive : UpdateInput
	MediaLive : UpdateInputDevice
	MediaLive : UpdateInputSecurityGroup
	MediaLive : UpdateMultiplex
	MediaLive : UpdateMultiplexProgram
	MediaLive : UpdateNetwork
	MediaLive : UpdateNode
	MediaLive : UpdateNodeState
	MediaLive : UpdateReservation
	MediaLive : UpdateSdiSource

Préfixe de service	Actions
mediatailor	tailleur multimédia : ConfigureLogsForPlaybackConfiguration tailleur multimédia : CreateChannel tailleur multimédia : CreateLiveSource tailleur multimédia : CreatePrefetchSchedule tailleur multimédia : CreateProgram tailleur multimédia : CreateSourceLocation tailleur multimédia : CreateVodSource tailleur multimédia : DeleteChannel tailleur multimédia : DeleteChannelPolicy tailleur multimédia : DeleteLiveSource tailleur multimédia : DeletePlaybackConfiguration tailleur multimédia : DeletePrefetchSchedule tailleur multimédia : DeleteProgram tailleur multimédia : DeleteSourceLocation tailleur multimédia : DeleteVodSource tailleur multimédia : DescribeChannel tailleur multimédia : DescribeLiveSource tailleur multimédia : DescribeProgram tailleur multimédia : DescribeSourceLocation tailleur multimédia : DescribeVodSource tailleur multimédia : GetChannelPolicy

Préfixe de service	Actions
	tailleur multimédia : GetChannelSchedule tailleur multimédia : GetPlaybackConfiguration tailleur multimédia : GetPrefetchSchedule tailleur multimédia : ListAlerts tailleur multimédia : ListChannels tailleur multimédia : ListLiveSources tailleur multimédia : ListPlaybackConfigurations tailleur multimédia : ListPrefetchSchedules tailleur multimédia : ListSourceLocations tailleur multimédia : ListVodSources tailleur multimédia : PutChannelPolicy tailleur multimédia : PutPlaybackConfiguration tailleur multimédia : StartChannel tailleur multimédia : StopChannel tailleur multimédia : UpdateChannel tailleur multimédia : UpdateLiveSource tailleur multimédia : UpdateProgram tailleur multimédia : UpdateSourceLocation tailleur multimédia : UpdateVodSource

Préfixe de service	Actions
base de données de mémoire	base de données de mémoire : BatchUpdateCluster
	base de données de mémoire : CopySnapshot
	base de données de mémoire : CreateAcl
	base de données de mémoire : CreateCluster
	base de données de mémoire : CreateMultiRegionCluster
	base de données de mémoire : CreateParameterGroup
	base de données de mémoire : CreateSnapshot
	base de données de mémoire : CreateSubnetGroup
	base de données de mémoire : CreateUser
	base de données de mémoire : DeleteAcl
	base de données de mémoire : DeleteCluster
	base de données de mémoire : DeleteMultiRegionCluster
	base de données de mémoire : DeleteParameterGroup
	base de données de mémoire : DeleteSnapshot
	base de données de mémoire : DeleteSubnetGroup
	base de données de mémoire : DeleteUser
	base de données de mémoire : DescribeAcls
	base de données de mémoire : DescribeClusters
	base de données de mémoire : DescribeEngineVersions
	base de données de mémoire : DescribeEvents
	base de données de mémoire : DescribeMultiRegionClusters

Préfixe de service	Actions
	base de données de mémoire : DescribeMultiRegionParameterGroups
	base de données de mémoire : DescribeMultiRegionParameters
	base de données de mémoire : DescribeParameterGroups
	base de données de mémoire : DescribeParameters
	base de données de mémoire : DescribeReservedNodes
	base de données de mémoire : DescribeReservedNodesOfferings
	base de données de mémoire : DescribeServiceUpdates
	base de données de mémoire : DescribeSnapshots
	base de données de mémoire : DescribeSubnetGroups
	base de données de mémoire : DescribeUsers
	base de données de mémoire : FailoverShard
	base de données de mémoire : ListAllowedMultiRegionClusterUpdates
	base de données de mémoire : ListAllowedNodeTypeUpdates
	base de données de mémoire : PurchaseReservedNodesOffering
	base de données de mémoire : ResetParameterGroup
	base de données de mémoire : UpdateAcl
	base de données de mémoire : UpdateCluster
	base de données de mémoire : UpdateMultiRegionCluster
	base de données de mémoire : UpdateParameterGroup
	base de données de mémoire : UpdateSubnetGroup

Préfixe de service	Actions
	base de données de mémoire : UpdateUser

Préfixe de service	Actions
mgh	mgh : AssociateCreatedArtifact mgh : AssociateDiscoveredResource mgh : AssociateSourceResource mgh : CreateHomeRegionControl mgh : CreateProgressUpdateStream mgh : DeleteHomeRegionControl mgh : DeleteProgressUpdateStream mgh : DescribeApplicationState mgh : DescribeHomeRegionControls mgh : DescribeMigrationTask mgh : DisassociateCreatedArtifact mgh : DisassociateDiscoveredResource mgh : DisassociateSourceResource mgh : GetHomeRegion mgh : ImportMigrationTask mgh : ListApplicationStates mgh : ListCreatedArtifacts mgh : ListDiscoveredResources mgh : ListMigrationTaskUpdates mgh : ListMigrationTasks mgh : ListProgressUpdateStreams

Préfixe de service	Actions
	mgh : ListSourceResources mgh : NotifyApplicationState mgh : NotifyMigrationTaskState mgh : PutResourceAttributes

Préfixe de service	Actions
mgn	mitrailleuse : ArchiveApplication
	mitrailleuse : ArchiveWave
	mitrailleuse : AssociateApplications
	mitrailleuse : AssociateSourceServers
	mitrailleuse : ChangeServerLifeCycleState
	mitrailleuse : CreateApplication
	mitrailleuse : CreateConnector
	mitrailleuse : CreateLaunchConfigurationTemplate
	mitrailleuse : CreateNetworkMigrationDefinition
	mitrailleuse : CreateReplicationConfigurationTemplate
	mitrailleuse : CreateWave
	mitrailleuse : DeleteApplication
	mitrailleuse : DeleteConnector
	mitrailleuse : DeleteJob
	mitrailleuse : DeleteLaunchConfigurationTemplate
	mitrailleuse : DeleteNetworkMigrationDefinition
	mitrailleuse : DeleteReplicationConfigurationTemplate
	mitrailleuse : DeleteSourceServer
	mitrailleuse : DeleteVcenterClient
	mitrailleuse : DeleteWave
	mitrailleuse : DescribeJobLogItems

Préfixe de service	Actions
	mitrailleuse : DescribeJobs
	mitrailleuse : DescribeLaunchConfigurationTemplates
	mitrailleuse : DescribeReplicationConfigurationTemplates
	mitrailleuse : DescribeVcenterClients
	mitrailleuse : DisassociateApplications
	mitrailleuse : DisassociateSourceServers
	mitrailleuse : DisconnectFromService
	mitrailleuse : FinalizeCutover
	mitrailleuse : GetNetworkMigrationDefinition
	mitrailleuse : GetNetworkMigrationMapperSegmentConstruct
	mitrailleuse : GetReplicationConfiguration
	mitrailleuse : InitializeService
	mitrailleuse : ListConnectors
	mitrailleuse : ListExportErrors
	mitrailleuse : ListExports
	mitrailleuse : ListImportErrors
	mitrailleuse : ListImportFileEnrichments
	mitrailleuse : ListImports
	mitrailleuse : ListManagedAccounts
	mitrailleuse : ListNetworkMigrationAnalyses
	mitrailleuse : ListNetworkMigrationAnalysisResults

Préfixe de service	Actions
	mitrailleuse : ListNetworkMigrationCodeGenerationSegments
	mitrailleuse : ListNetworkMigrationCodeGenerations
	mitrailleuse : ListNetworkMigrationDefinitions
	mitrailleuse : ListNetworkMigrationDeployedStacks
	mitrailleuse : ListNetworkMigrationDeployments
	mitrailleuse : ListNetworkMigrationExecutions
	mitrailleuse : ListNetworkMigrationMapperSegmentConstructs
	mitrailleuse : ListNetworkMigrationMapperSegments
	mitrailleuse : ListNetworkMigrationMappingUpdates
	mitrailleuse : ListNetworkMigrationMappings
	mitrailleuse : ListSourceServerActions
	mitrailleuse : ListTemplateActions
	mitrailleuse : MarkAsArchived
	mitrailleuse : PauseReplication
	mitrailleuse : PutSourceServerAction
	mitrailleuse : PutTemplateAction
	mitrailleuse : RemoveSourceServerAction
	mitrailleuse : RemoveTemplateAction
	mitrailleuse : ResumeReplication
	mitrailleuse : RetryDataReplication
	mitrailleuse : StartCutover

Préfixe de service	Actions
	mitrailleuse : StartExport
	mitrailleuse : StartImport
	mitrailleuse : StartImportFileEnrichment
	mitrailleuse : StartNetworkMigrationCodeGeneration
	mitrailleuse : StartNetworkMigrationMapping
	mitrailleuse : StartNetworkMigrationMappingUpdate
	mitrailleuse : StartReplication
	mitrailleuse : StartTest
	mitrailleuse : StopReplication
	mitrailleuse : TerminateTargetInstances
	mitrailleuse : UnarchiveApplication
	mitrailleuse : UnarchiveWave
	mitrailleuse : UpdateApplication
	mitrailleuse : UpdateConnector
	mitrailleuse : UpdateLaunchConfigurationTemplate
	mitrailleuse : UpdateNetworkMigrationDefinition
	mitrailleuse : UpdateNetworkMigrationMapperSegment
	mitrailleuse : UpdateReplicationConfiguration
	mitrailleuse : UpdateReplicationConfigurationTemplate
	mitrailleuse : UpdateSourceServer
	mitrailleuse : UpdateSourceServerReplicationType

Préfixe de service	Actions
	mitrailleuse : UpdateWave

Préfixe de service	Actions
migrationhub-strategy	stratégie du pôle migratoire : GetAntiPattern stratégie du pôle migratoire : GetApplicationComponentDetails stratégie du pôle migratoire : GetApplicationComponentStrategies stratégie du pôle migratoire : GetAssessment stratégie du pôle migratoire : GetImportFileTask stratégie du pôle migratoire : GetLatestAssessmentId stratégie du pôle migratoire : GetMessage stratégie du pôle migratoire : GetPortfolioPreferences stratégie du pôle migratoire : GetPortfolioSummary stratégie du pôle migratoire : GetRecommendationReportDetails stratégie du pôle migratoire : GetServerDetails stratégie du pôle migratoire : GetServerStrategies stratégie du pôle migratoire : ListAnalyzableServers stratégie du pôle migratoire : ListAntiPatterns stratégie du pôle migratoire : ListApplicationComponents stratégie du pôle migratoire : ListCollectors stratégie du pôle migratoire : ListImportFileTask stratégie du pôle migratoire : ListJarArtifacts stratégie du pôle migratoire : ListServers stratégie du pôle migratoire : PutLogData stratégie du pôle migratoire : PutMetricData

Préfixe de service	Actions
	stratégie du pôle migratoire : PutPortfolioPreferences stratégie du pôle migratoire : RegisterCollector stratégie du pôle migratoire : SendMessage stratégie du pôle migratoire : StartAssessment stratégie du pôle migratoire : StartImportFileTask stratégie du pôle migratoire : StartRecommendationReportGeneration stratégie du pôle migratoire : StopAssessment stratégie du pôle migratoire : UpdateApplicationComponentConfig stratégie du pôle migratoire : UpdateCollectorConfiguration stratégie du pôle migratoire : UpdateServerConfig

Préfixe de service	Actions
mobiletargeting	ciblage mobile : CreateApp
	ciblage mobile : CreateCampaign
	ciblage mobile : CreateEmailTemplate
	ciblage mobile : CreateExportJob
	ciblage mobile : CreateImportJob
	ciblage mobile : CreateInAppTemplate
	ciblage mobile : CreateJourney
	ciblage mobile : CreatePushTemplate
	ciblage mobile : CreateRecommenderConfiguration
	ciblage mobile : CreateSegment
	ciblage mobile : CreateSmsTemplate
	ciblage mobile : CreateVoiceTemplate
	ciblage mobile : DeleteAdmChannel
	ciblage mobile : DeleteApnsChannel
	ciblage mobile : DeleteApnsSandboxChannel
	ciblage mobile : DeleteApnsVoipChannel
	ciblage mobile : DeleteApnsVoipSandboxChannel
	ciblage mobile : DeleteApp
	ciblage mobile : DeleteBaiduChannel
	ciblage mobile : DeleteCampaign
	ciblage mobile : DeleteEmailChannel

Préfixe de service	Actions
	ciblage mobile : DeleteEmailTemplate
	ciblage mobile : DeleteEndpoint
	ciblage mobile : DeleteEventStream
	ciblage mobile : DeleteGcmChannel
	ciblage mobile : DeleteInAppTemplate
	ciblage mobile : DeleteJourney
	ciblage mobile : DeletePushTemplate
	ciblage mobile : DeleteRecommenderConfiguration
	ciblage mobile : DeleteSegment
	ciblage mobile : DeleteSmsChannel
	ciblage mobile : DeleteSmsTemplate
	ciblage mobile : DeleteUserEndpoints
	ciblage mobile : DeleteVoiceChannel
	ciblage mobile : DeleteVoiceTemplate
	ciblage mobile : GetAdmChannel
	ciblage mobile : GetApnsChannel
	ciblage mobile : GetApnsSandboxChannel
	ciblage mobile : GetApnsVoipChannel
	ciblage mobile : GetApnsVoipSandboxChannel
	ciblage mobile : GetApp
	ciblage mobile : GetApplicationDateRangeKpi

Préfixe de service	Actions
	ciblage mobile : GetApplicationSettings
	ciblage mobile : GetApps
	ciblage mobile : GetBaiduChannel
	ciblage mobile : GetCampaign
	ciblage mobile : GetCampaignActivities
	ciblage mobile : GetCampaignDateRangeKpi
	ciblage mobile : GetCampaignVersion
	ciblage mobile : GetCampaignVersions
	ciblage mobile : GetCampaigns
	ciblage mobile : GetChannels
	ciblage mobile : GetEmailChannel
	ciblage mobile : GetEmailTemplate
	ciblage mobile : GetEndpoint
	ciblage mobile : GetEventStream
	ciblage mobile : GetExportJob
	ciblage mobile : GetExportJobs
	ciblage mobile : GetGcmChannel
	ciblage mobile : GetImportJob
	ciblage mobile : GetImportJobs
	ciblage mobile : GetInAppMessages
	ciblage mobile : GetInAppTemplate

Préfixe de service	Actions
	ciblage mobile : GetJourney
	ciblage mobile : GetJourneyDateRangeKpi
	ciblage mobile : GetJourneyExecutionActivityMetrics
	ciblage mobile : GetJourneyExecutionMetrics
	ciblage mobile : GetJourneyRunExecutionActivityMetrics
	ciblage mobile : GetJourneyRunExecutionMetrics
	ciblage mobile : GetJourneyRuns
	ciblage mobile : GetPushTemplate
	ciblage mobile : GetRecommenderConfiguration
	ciblage mobile : GetRecommenderConfigurations
	ciblage mobile : GetSegment
	ciblage mobile : GetSegmentExportJobs
	ciblage mobile : GetSegmentImportJobs
	ciblage mobile : GetSegmentVersion
	ciblage mobile : GetSegmentVersions
	ciblage mobile : GetSegments
	ciblage mobile : GetSmsChannel
	ciblage mobile : GetSmsTemplate
	ciblage mobile : GetUserEndpoints
	ciblage mobile : GetVoiceChannel
	ciblage mobile : GetVoiceTemplate

Préfixe de service	Actions
	ciblage mobile : ListJourneys
	ciblage mobile : ListTemplateVersions
	ciblage mobile : ListTemplates
	ciblage mobile : PhoneNumberValidate
	ciblage mobile : PutEventStream
	ciblage mobile : RemoveAttributes
	ciblage mobile : UpdateAdmChannel
	ciblage mobile : UpdateApnsChannel
	ciblage mobile : UpdateApnsSandboxChannel
	ciblage mobile : UpdateApnsVoipChannel
	ciblage mobile : UpdateApnsVoipSandboxChannel
	ciblage mobile : UpdateApplicationSettings
	ciblage mobile : UpdateBaiduChannel
	ciblage mobile : UpdateCampaign
	ciblage mobile : UpdateEmailChannel
	ciblage mobile : UpdateEmailTemplate
	ciblage mobile : UpdateEndpoint
	ciblage mobile : UpdateEndpointsBatch
	ciblage mobile : UpdateGcmChannel
	ciblage mobile : UpdateInAppTemplate
	ciblage mobile : UpdateJourney

Préfixe de service	Actions
	ciblage mobile : UpdateJourneyState
	ciblage mobile : UpdatePushTemplate
	ciblage mobile : UpdateRecommenderConfiguration
	ciblage mobile : UpdateSegment
	ciblage mobile : UpdateSmsChannel
	ciblage mobile : UpdateSmsTemplate
	ciblage mobile : UpdateTemplateActiveVersion
	ciblage mobile : UpdateVoiceChannel
	ciblage mobile : UpdateVoiceTemplate
	mobiletargeting:VerifyOTPMessage

Préfixe de service	Actions
mq	mq : CreateBroker mq : CreateConfiguration mq : CreateUser mq : DeleteBroker mq : DeleteConfiguration mq : DeleteUser mq : DescribeBroker mq : DescribeBrokerEngineTypes mq : DescribeBrokerInstanceOptions mq : DescribeConfiguration mq : DescribeConfigurationRevision mq : DescribeSharedResources mq : DescribeUser mq : ListBrokers mq : ListConfigurationRevisions mq : ListConfigurations mq : ListUsers mq:Promote mq : RebootBroker mq : UpdateBroker mq : UpdateConfiguration

Préfixe de service	Actions
	mq : UpdateUser

Préfixe de service	Actions
networkmanager	gestionnaire de réseau : AcceptAttachment
	gestionnaire de réseau : AssociateConnectPeer
	gestionnaire de réseau : AssociateCustomerGateway
	gestionnaire de réseau : AssociateLink
	gestionnaire de réseau : AssociateTransitGatewayConnectPeer
	gestionnaire de réseau : CreateConnectAttachment
	gestionnaire de réseau : CreateConnectPeer
	gestionnaire de réseau : CreateConnection
	gestionnaire de réseau : CreateCoreNetwork
	gestionnaire de réseau : CreateDevice
	gestionnaire de réseau : CreateDirectConnectGatewayAttachment
	gestionnaire de réseau : CreateGlobalNetwork
	gestionnaire de réseau : CreateLink
	gestionnaire de réseau : CreateSite
	gestionnaire de réseau : CreateSiteToSiteVpnAttachment
	gestionnaire de réseau : CreateTransitGatewayPeering
	gestionnaire de réseau : CreateTransitGatewayRouteTableAttachment
	gestionnaire de réseau : CreateVpcAttachment
	gestionnaire de réseau : DeleteAttachment
	gestionnaire de réseau : DeleteConnectPeer

Préfixe de service	Actions
	gestionnaire de réseau : DeleteConnection
	gestionnaire de réseau : DeleteCoreNetwork
	gestionnaire de réseau : DeleteCoreNetworkPolicyVersion
	gestionnaire de réseau : DeleteDevice
	gestionnaire de réseau : DeleteGlobalNetwork
	gestionnaire de réseau : DeleteLink
	gestionnaire de réseau : DeletePeering
	gestionnaire de réseau : DeleteResourcePolicy
	gestionnaire de réseau : DeleteSite
	gestionnaire de réseau : DeregisterTransitGateway
	gestionnaire de réseau : DescribeGlobalNetworks
	gestionnaire de réseau : DisassociateConnectPeer
	gestionnaire de réseau : DisassociateCustomerGateway
	gestionnaire de réseau : DisassociateLink
	gestionnaire de réseau : DisassociateTransitGatewayConnectPeer
	gestionnaire de réseau : ExecuteCoreNetworkChangeSet
	gestionnaire de réseau : GetConnectAttachment
	gestionnaire de réseau : GetConnectPeer
	gestionnaire de réseau : GetConnectPeerAssociations
	gestionnaire de réseau : GetConnections
	gestionnaire de réseau : GetCoreNetwork

Préfixe de service	Actions
	gestionnaire de réseau : GetCoreNetworkChangeEvents
	gestionnaire de réseau : GetCoreNetworkChangeSet
	gestionnaire de réseau : GetCoreNetworkPolicy
	gestionnaire de réseau : GetCustomerGatewayAssociations
	gestionnaire de réseau : GetDevices
	gestionnaire de réseau : GetLinkAssociations
	gestionnaire de réseau : GetLinks
	gestionnaire de réseau : GetNetworkResourceCounts
	gestionnaire de réseau : GetNetworkResourceRelationships
	gestionnaire de réseau : GetNetworkResources
	gestionnaire de réseau : GetNetworkRoutes
	gestionnaire de réseau : GetNetworkTelemetry
	gestionnaire de réseau : GetResourcePolicy
	gestionnaire de réseau : GetRouteAnalysis
	gestionnaire de réseau : GetSiteToSiteVpnAttachment
	gestionnaire de réseau : GetSites
	gestionnaire de réseau : GetTransitGatewayConnectPeerAssociations
	gestionnaire de réseau : GetTransitGatewayPeering
	gestionnaire de réseau : GetTransitGatewayRegistrations
	gestionnaire de réseau : GetTransitGatewayRouteTableAttachment

Préfixe de service	Actions
	gestionnaire de réseau : GetVpcAttachment
	gestionnaire de réseau : ListAttachmentRoutingPolicyAssociations
	gestionnaire de réseau : ListAttachments
	gestionnaire de réseau : ListConnectPeers
	gestionnaire de réseau : ListCoreNetworkPolicyVersions
	gestionnaire de réseau : ListCoreNetworkPrefixListAssociations
	gestionnaire de réseau : ListCoreNetworkRoutingInformation
	gestionnaire de réseau : ListCoreNetworks
	gestionnaire de réseau : ListOrganizationServiceAccessStatus
	gestionnaire de réseau : ListPeerings
	gestionnaire de réseau : PutAttachmentRoutingPolicyLabel
	gestionnaire de réseau : PutCoreNetworkPolicy
	gestionnaire de réseau : PutResourcePolicy
	gestionnaire de réseau : RegisterTransitGateway
	gestionnaire de réseau : RejectAttachment
	gestionnaire de réseau : RemoveAttachmentRoutingPolicyLabel
	gestionnaire de réseau : RestoreCoreNetworkPolicyVersion
	gestionnaire de réseau : StartOrganizationServiceAccessUpdate
	gestionnaire de réseau : StartRouteAnalysis
	gestionnaire de réseau : UpdateConnection
	gestionnaire de réseau : UpdateCoreNetwork

Préfixe de service	Actions
	gestionnaire de réseau : UpdateDevice
	gestionnaire de réseau : UpdateDirectConnectGatewayAttachment
	gestionnaire de réseau : UpdateGlobalNetwork
	gestionnaire de réseau : UpdateLink
	gestionnaire de réseau : UpdateNetworkResourceMetadata
	gestionnaire de réseau : UpdateSite
	gestionnaire de réseau : UpdateVpcAttachment

Préfixe de service	Actions
nimble	agile : AcceptEulas agile : CreateLaunchProfile agile : CreateStreamingImage agile : CreateStreamingSession agile : CreateStreamingSessionStream agile : CreateStudio agile : CreateStudioComponent agile : DeleteLaunchProfile agile : DeleteLaunchProfileMember agile : DeleteStreamingImage agile : DeleteStreamingSession agile : DeleteStudio agile : DeleteStudioComponent agile : DeleteStudioMember agile : GetEula agile : GetLaunchProfileDetails agile : GetStreamingImage agile : GetStreamingSession agile : GetStreamingSessionBackup agile : GetStreamingSessionStream agile : GetStudio

Préfixe de service	Actions
	agile : GetStudioComponent agile : GetStudioMember agile : ListEulas agile : ListLaunchProfileMembers agile : ListLaunchProfiles agile : ListStreamingImages agile : ListStreamingSessionBackups agile : ListStreamingSessions agile : ListStudioComponents agile : ListStudioMembers agile : ListStudios agile : PutLaunchProfileMembers agile : PutStudioMembers agile : StartStreamingSession agile : StartStudioSSOConfigurationRepair agile : StopStreamingSession agile : UpdateLaunchProfile agile : UpdateLaunchProfileMember agile : UpdateStreamingImage agile : UpdateStudio agile : UpdateStudioComponent

Préfixe de service	Actions
omics	bandes dessinées : AbortMultipartReadSetUpload
	bandes dessinées : AcceptShare
	bandes dessinées : BatchDeleteReadSet
	bandes dessinées : CancelAnnotationImportJob
	bandes dessinées : CancelRun
	bandes dessinées : CancelRunBatch
	bandes dessinées : CancelVariantImportJob
	bandes dessinées : CompleteMultipartReadSetUpload
	bandes dessinées : CreateAnnotationStore
	bandes dessinées : CreateAnnotationStoreVersion
	bandes dessinées : CreateConfiguration
	bandes dessinées : CreateMultipartReadSetUpload
	bandes dessinées : CreateReferenceStore
	bandes dessinées : CreateRunGroup
	bandes dessinées : CreateSequenceStore
	bandes dessinées : CreateShare
	bandes dessinées : CreateVariantStore
	bandes dessinées : CreateWorkflow
	bandes dessinées : CreateWorkflowVersion
	bandes dessinées : DeleteAnnotationStore
	bandes dessinées : DeleteAnnotationStoreVersions

Préfixe de service	Actions
	bandes dessinées : DeleteBatch
	bandes dessinées : DeleteConfiguration
	bandes dessinées : DeleteReference
	bandes dessinées : DeleteReferenceStore
	bandes dessinées : DeleteRun
	bandes dessinées : DeleteRunBatch
	bandes dessinées : DeleteRunGroup
	bandes dessinées : DeleteSequenceStore
	bandes dessinées : DeleteShare
	bandes dessinées : DeleteVariantStore
	bandes dessinées : DeleteWorkflow
	bandes dessinées : DeleteWorkflowVersion
	bandes dessinées : GetAnnotationImportJob
	bandes dessinées : GetAnnotationStore
	bandes dessinées : GetAnnotationStoreVersion
	bandes dessinées : GetBatch
	bandes dessinées : GetConfiguration
	bandes dessinées : GetReadSet
	bandes dessinées : GetReadSetActivationJob
	bandes dessinées : GetReadSetExportJob
	bandes dessinées : GetReadSetImportJob

Préfixe de service	Actions
	bandes dessinées : GetReadSetMetadata
	bandes dessinées : GetReference
	bandes dessinées : GetReferenceImportJob
	bandes dessinées : GetReferenceMetadata
	bandes dessinées : GetReferenceStore
	bandes dessinées : GetRun
	bandes dessinées : GetRunGroup
	bandes dessinées : GetRunTask
	bandes dessinées : GetSequenceStore
	bandes dessinées : GetShare
	bandes dessinées : GetVariantImportJob
	bandes dessinées : GetVariantStore
	bandes dessinées : GetWorkflow
	bandes dessinées : GetWorkflowVersion
	bandes dessinées : ListAnnotationImportJobs
	bandes dessinées : ListAnnotationStoreVersions
	bandes dessinées : ListAnnotationStores
	bandes dessinées : ListBatch
	bandes dessinées : ListConfigurations
	bandes dessinées : ListMultipartReadSetUploads
	bandes dessinées : ListReadSetActivationJobs

Préfixe de service	Actions
	bandes dessinées : ListReadSetExportJobs
	bandes dessinées : ListReadSetImportJobs
	bandes dessinées : ListReadSetUploadParts
	bandes dessinées : ListReadSets
	bandes dessinées : ListReferenceImportJobs
	bandes dessinées : ListReferenceStores
	bandes dessinées : ListReferences
	bandes dessinées : ListRunGroups
	bandes dessinées : ListRunTasks
	bandes dessinées : ListRuns
	bandes dessinées : ListRunsInBatch
	bandes dessinées : ListSequenceStores
	bandes dessinées : ListShares
	bandes dessinées : ListVariantImportJobs
	bandes dessinées : ListVariantStores
	bandes dessinées : ListWorkflowVersions
	bandes dessinées : ListWorkflows
	bandes dessinées : StartAnnotationImportJob
	bandes dessinées : StartReadSetActivationJob
	bandes dessinées : StartReadSetExportJob
	bandes dessinées : StartReadSetImportJob

Préfixe de service	Actions
	bandes dessinées : StartReferenceImportJob
	bandes dessinées : StartRun
	bandes dessinées : StartRunBatch
	bandes dessinées : StartVariantImportJob
	bandes dessinées : UpdateAnnotationStore
	bandes dessinées : UpdateAnnotationStoreVersion
	bandes dessinées : UpdateRunGroup
	bandes dessinées : UpdateVariantStore
	bandes dessinées : UpdateWorkflow
	bandes dessinées : UpdateWorkflowVersion
	bandes dessinées : UploadReadSetPart

Préfixe de service	Actions
opsworks	opsworks : AssignInstance opsworks : AssignVolume opsworks : AssociateElasticIp opsworks : AttachElasticLoadBalancer opsworks : CloneStack opsworks : CreateApp opsworks : CreateDeployment opsworks : CreateInstance opsworks : CreateLayer opsworks : CreateStack opsworks : CreateUserProfile opsworks : DeleteApp opsworks : DeleteInstance opsworks : DeleteLayer opsworks : DeleteStack opsworks : DeleteUserProfile opsworks : DeregisterEcsCluster opsworks : DeregisterElasticIp opsworks : DeregisterInstance opsworks : DeregisterRdsDbInstance opsworks : DeregisterVolume

Préfixe de service	Actions
	opsworks : DescribeAgentVersions opsworks : DescribeApps opsworks : DescribeCommands opsworks : DescribeDeployments opsworks : DescribeEcsClusters opsworks : DescribeElasticIps opsworks : DescribeElasticLoadBalancers opsworks : DescribeInstances opsworks : DescribeLayers opsworks : DescribeLoadBasedAutoScaling opsworks : DescribeMyUserProfile opsworks : DescribeOperatingSystems opsworks : DescribePermissions opsworks : DescribeRaidArrays opsworks : DescribeRdsDbInstances opsworks : DescribeServiceErrors opsworks : DescribeStackProvisioningParameters opsworks : DescribeStackSummary opsworks : DescribeStacks opsworks : DescribeTimeBasedAutoScaling opsworks : DescribeUserProfiles

Préfixe de service	Actions
	opsworks : DescribeVolumes opsworks : DetachElasticLoadBalancer opsworks : DisassociateElasticIp opsworks : GetHostnameSuggestion opsworks : GrantAccess opsworks : RebootInstance opsworks : RegisterEcsCluster opsworks : RegisterElasticIp opsworks : RegisterInstance opsworks : RegisterRdsDbInstance opsworks : RegisterVolume opsworks : SetLoadBasedAutoScaling opsworks : SetPermission opsworks : SetTimeBasedAutoScaling opsworks : StartInstance opsworks : StartStack opsworks : StopInstance opsworks : StopStack opsworks : UnassignInstance opsworks : UnassignVolume opsworks : UpdateApp

Préfixe de service	Actions
	opsworks : UpdateElasticIp opsworks : UpdateInstance opsworks : UpdateLayer opsworks : UpdateMyUserProfile opsworks : UpdateRdsDbInstance opsworks : UpdateStack opsworks : UpdateUserProfile opsworks : UpdateVolume

Préfixe de service	Actions
opsworks-cm	Opsworks-cm : AssociateNode
	Opsworks-cm : CreateBackup
	Opsworks-cm : CreateServer
	Opsworks-cm : DeleteBackup
	Opsworks-cm : DeleteServer
	Opsworks-cm : DescribeAccountAttributes
	Opsworks-cm : DescribeBackups
	Opsworks-cm : DescribeEvents
	Opsworks-cm : DescribeNodeAssociationStatus
	Opsworks-cm : DescribeServers
	Opsworks-cm : DisassociateNode
	Opsworks-cm : ExportServerEngineAttribute
	Opsworks-cm : RestoreServer
	Opsworks-cm : StartMaintenance
	Opsworks-cm : UpdateServer
	Opsworks-cm : UpdateServerEngineAttributes

Préfixe de service	Actions
organisations	organisations : AcceptHandshake organisations : AttachPolicy organisations : CancelHandshake organisations : CloseAccount organisations : CreateAccount organisations : CreateGovCloudAccount organisations : CreateOrganization organisations : CreateOrganizationalUnit organisations : CreatePolicy organisations : DeclineHandshake organisations : DeleteOrganization organisations : DeleteOrganizationalUnit organisations : DeletePolicy organisations : DeleteResourcePolicy organisations : DeregisterDelegatedAdministrator organisations : DescribeAccount organisations : DescribeCreateAccountStatus organisations : DescribeEffectivePolicy organisations : DescribeHandshake organisations : DescribeOrganization organisations : DescribeOrganizationalUnit

Préfixe de service	Actions
	organisations : DescribePolicy organisations : DescribeResourcePolicy organisations : DescribeResponsibilityTransfer organisations : DetachPolicy organisations : DisableAWSServiceAccess organisations : DisablePolicyType organisations : EnableAWSServiceAccess organisations : EnableAllFeatures organisations : EnablePolicyType organisations : InviteAccountToOrganization organisations : LeaveOrganization organisations : ListAWSServiceAccessForOrganization organisations : ListAccounts organisations : ListAccountsForParent organisations : ListAccountsWithInvalidEffectivePolicy organisations : ListChildren organisations : ListCreateAccountStatus organisations : ListDelegatedAdministrators organisations : ListDelegatedServicesForAccount organisations : ListHandshakesForAccount organisations : ListHandshakesForOrganization

Préfixe de service	Actions
	organisations : ListInboundResponsibilityTransfers organisations : ListOrganizationalUnitsForParent organisations : ListOutboundResponsibilityTransfers organisations : ListParents organisations : ListPolicies organisations : ListPoliciesForTarget organisations : ListRoots organisations : ListTargetsForPolicy organisations : MoveAccount organisations : PutResourcePolicy organisations : RegisterDelegatedAdministrator organisations : RemoveAccountFromOrganization organisations : TerminateResponsibilityTransfer organisations : UpdateOrganizationalUnit organisations : UpdatePolicy organisations : UpdateResponsibilityTransfer

Préfixe de service	Actions
outposts	avant-postes : CancelCapacityTask
	avant-postes : CancelOrder
	avant-postes : CreateOrder
	avant-postes : CreateOutpost
	avant-postes : CreatePrivateConnectivityConfig
	avant-postes : CreateQuote
	avant-postes : CreateRenewal
	avant-postes : CreateSite
	avant-postes : DeleteOutpost
	avant-postes : DeleteQuote
	avant-postes : DeleteSite
	avant-postes : GetCapacityTask
	avant-postes : GetCatalogItem
	avant-postes : GetConnection
	avant-postes : GetOrder
	avant-postes : GetOutpost
	avant-postes : GetOutpostBillingInformation
	avant-postes : GetOutpostInstanceTypes
	avant-postes : GetOutpostSupportedInstanceTypes
	avant-postes : GetPrivateConnectivityConfig
	avant-postes : GetQuote

Préfixe de service	Actions
	avant-postes : GetRenewalPricing
	avant-postes : GetSite
	avant-postes : GetSiteAddress
	avant-postes : ListAssetInstances
	avant-postes : ListAssets
	avant-postes : ListBlockingInstancesForCapacityTask
	avant-postes : ListCapacityTasks
	avant-postes : ListCatalogItems
	avant-postes : ListOrderableInstanceTypes
	avant-postes : ListOrders
	avant-postes : ListOutposts
	avant-postes : ListQuotes
	avant-postes : ListSites
	avant-postes : StartCapacityTask
	avant-postes : StartConnection
	avant-postes : UpdateOutpost
	avant-postes : UpdateQuote
	avant-postes : UpdateSite
	avant-postes : UpdateSiteAddress
	avant-postes : UpdateSiteRackPhysicalProperties

Préfixe de service	Actions
pi	épi : CreatePerformanceAnalysisReport épi : DeletePerformanceAnalysisReport épi : DescribeDimensionKeys épi : GetDimensionKeyDetails épi : GetPerformanceAnalysisReport épi : GetResourceMetadata épi : GetResourceMetrics épi : ListAvailableResourceDimensions épi : ListAvailableResourceMetrics épi : ListPerformanceAnalysisReports
pipes	tuyaux : CreatePipe tuyaux : DeletePipe tuyaux : DescribePipe tuyaux : ListPipes tuyaux : StartPipe tuyaux : StopPipe tuyaux : UpdatePipe

Préfixe de service	Actions
polly	Polly : DeleteLexicon Polly : DescribeVoices Polly : GetLexicon Polly : GetSpeechSynthesisTask Polly : ListLexicons Polly : ListSpeechSynthesisTasks Polly : PutLexicon Polly : StartSpeechSynthesisTask Polly : SynthesizeSpeech

Préfixe de service	Actions
profile	profil : AddProfileKey
	profil : BatchGetCalculatedAttributeForProfile
	profil : BatchGetProfile
	profil : CreateCalculatedAttributeDefinition
	profil : CreateDomain
	profil : CreateEventStream
	profil : CreateProfile
	profil : CreateRecommender
	profil : CreateRecommenderFilter
	profil : CreateRecommenderSchema
	profil : CreateSegmentDefinition
	profil : CreateSegmentEstimate
	profil : CreateSegmentSnapshot
	profil : CreateUploadJob
	profil : DeleteCalculatedAttributeDefinition
	profil : DeleteDomain
	profil : DeleteDomainObjectType
	profil : DeleteEventStream
	profil : DeleteIntegration
	profil : DeleteProfile
	profil : DeleteProfileKey

Préfixe de service	Actions
	profil : DeleteProfileObject
	profil : DeleteProfileObjectType
	profil : DeleteRecommender
	profil : DeleteRecommenderFilter
	profil : DeleteRecommenderSchema
	profil : DeleteSegmentDefinition
	profil : DeleteWorkflow
	profil : DetectProfileObjectType
	profil : GetAutoMergingPreview
	profil : GetCalculatedAttributeDefinition
	profil : GetCalculatedAttributeForProfile
	profil : GetDomain
	profil : GetDomainObjectType
	profil : GetEventStream
	profil : GetIdentityResolutionJob
	profil : GetIntegration
	profil : GetMatches
	profil : GetObjectTypeAttributeStatistics
	profil : GetProfileObjectType
	profil : GetProfileObjectTypeTemplate
	profil : GetProfileRecommendations

Préfixe de service	Actions
	profil : GetRecommender
	profil : GetRecommenderFilter
	profil : GetRecommenderSchema
	profil : GetSegmentDefinition
	profil : GetSegmentEstimate
	profil : GetSegmentMembership
	profil : GetSegmentSnapshot
	profil : GetSimilarProfiles
	profil : GetUploadJob
	profil : GetUploadJobPath
	profil : GetWorkflow
	profil : GetWorkflowSteps
	profil : ListAccountIntegrations
	profil : ListCalculatedAttributeDefinitions
	profil : ListCalculatedAttributesForProfile
	profil : ListDomainLayouts
	profil : ListDomainObjectTypes
	profil : ListDomains
	profil : ListEventStreams
	profil : ListIdentityResolutionJobs
	profil : ListIntegrations

Préfixe de service	Actions
	profil : ListObjectTypeAttributeValues
	profil : ListObjectTypeAttributes
	profil : ListProfileAttributeValues
	profil : ListProfileObjectTypeTemplates
	profil : ListProfileObjectTypes
	profil : ListProfileObjects
	profil : ListRecommenderFilters
	profil : ListRecommenderRecipes
	profil : ListRecommenderSchemas
	profil : ListRecommenders
	profil : ListRuleBasedMatches
	profil : ListSegmentDefinitions
	profil : ListUploadJobs
	profil : ListWorkflows
	profil : MergeProfiles
	profil : PutDomainObjectType
	profil : PutIntegration
	profil : PutProfileObject
	profil : PutProfileObjectType
	profil : SearchProfiles
	profil : StartRecommender

Préfixe de service	Actions
	profil : StartUploadJob
	profil : StopRecommender
	profil : StopUploadJob
	profil : UpdateCalculatedAttributeDefinition
	profil : UpdateDomain
	profil : UpdateProfile
	profil : UpdateRecommender

Préfixe de service	Actions
qldb	qldb : CancelJournalKinesisStream qldb : CreateLedger qldb : DeleteLedger qldb : DescribeJournalKinesisStream qldb : DescribeJournalS3Export qldb : DescribeLedger qldb : ExportJournalToS3 qldb : GetBlock qldb : GetDigest qldb : GetRevision qldb : ListJournalKinesisStreamsForLedger qldb : ListJournalS3Exports qldb : ListJournalS3ExportsForLedger qldb : ListLedgers qldb : StreamJournalToKinesis qldb : UpdateLedger qldb : UpdateLedgerPermissionsMode

Préfixe de service	Actions
ram	RAM : AcceptResourceShareInvitation
	RAM : AssociateResourceShare
	RAM : AssociateResourceSharePermission
	RAM : CreatePermission
	RAM : CreatePermissionVersion
	RAM : CreateResourceShare
	RAM : DeletePermission
	RAM : DeletePermissionVersion
	RAM : DeleteResourceShare
	RAM : DisassociateResourceShare
	RAM : DisassociateResourceSharePermission
	RAM : EnableSharingWithAwsOrganization
	RAM : GetPermission
	RAM : GetResourcePolicies
	RAM : GetResourceShareAssociations
	RAM : GetResourceShareInvitations
	RAM : GetResourceShares
	RAM : ListPendingInvitationResources
	RAM : ListPermissionAssociations
	RAM : ListPermissionVersions
	RAM : ListPermissions

Préfixe de service	Actions
	RAM : ListPrincipals RAM : ListReplacePermissionAssociationsWork RAM : ListResourceSharePermissions RAM : ListResourceTypes RAM : ListResources RAM : ListSourceAssociations RAM : PromotePermissionCreatedFromPolicy RAM : PromoteResourceShareCreatedFromPolicy RAM : RejectResourceShareInvitation RAM : ReplacePermissionAssociations RAM : SetDefaultPermissionVersion RAM : UpdateResourceShare
rbin	corbeille : CreateRule corbeille : DeleteRule corbeille : GetRule corbeille : ListRules corbeille : LockRule corbeille : UnlockRule corbeille : UpdateRule

Préfixe de service	Actions
rds	rouge : AddRoleToDBCluster rouge : AddRoleToDBInstance rouge : AddSourceIdentifierToSubscription rouge : ApplyPendingMaintenanceAction rouge : AuthorizeDBSecurityGroupIngress rds:BacktrackDBCluster rouge : CancelExportTask rouge : CopyDBClusterParameterGroup rouge : CopyDBClusterSnapshot rouge : CopyDBParameterGroup rds:CopyDBSnapshot rouge : CopyOptionGroup rouge : CreateCustomDBEngineVersion rouge : CreateDBClusterParameterGroup rouge : CreateDBParameterGroup rds:CreateDBProxy rouge : CreateDBProxyEndpoint rouge : CreateDBSecurityGroup rouge : CreateDBSubnetGroup rouge : CreateEventSubscription rouge : CreateGlobalCluster

Préfixe de service	Actions
	rouge : CreateOptionGroup
	rouge : DeleteBlueGreenDeployment
	rouge : DeleteDBClusterAutomatedBackup
	rouge : DeleteDBClusterParameterGroup
	rouge : DeleteDBClusterSnapshot
	rouge : DeleteDBInstanceAutomatedBackup
	rouge : DeleteDBParameterGroup
	rds:DeleteDBProxy
	rouge : DeleteDBProxyEndpoint
	rouge : DeleteDBSecurityGroup
	rds:DeleteDBSnapshot
	rouge : DeleteDBSubnetGroup
	rouge : DeleteEventSubscription
	rouge : DeleteGlobalCluster
	rouge : DeleteOptionGroup
	rouge : DeregisterDBProxyTargets
	rouge : DescribeAccountAttributes
	rouge : DescribeBlueGreenDeployments
	rouge : DescribeCertificates
	rouge : DescribeDBClusterAutomatedBackups
	rouge : DescribeDBClusterBacktracks

Préfixe de service	Actions
	rouge : DescribeDBClusterEndpoints
	rouge : DescribeDBClusterParameterGroups
	rouge : DescribeDBClusterParameters
	rouge : DescribeDBClusterSnapshotAttributes
	rouge : DescribeDBClusterSnapshots
	rds:DescribeDBClusters
	rouge : DescribeDBEngineVersions
	rouge : DescribeDBInstanceAutomatedBackups
	rds:DescribeDBInstances
	rouge : DescribeDBLogFiles
	rouge : DescribeDBMajorEngineVersions
	rouge : DescribeDBParameterGroups
	rds:DescribeDBParameters
	rds:DescribeDBProxies
	rouge : DescribeDBProxyEndpoints
	rouge : DescribeDBProxyTargetGroups
	rouge : DescribeDBProxyTargets
	rds:DescribeDBRecommendations
	rouge : DescribeDBSecurityGroups
	rouge : DescribeDBSnapshotAttributes
	rouge : DescribeDBSnapshotTenantDatabases

Préfixe de service	Actions
	rds:DescribeDBSnapshots rouge : DescribeDBSubnetGroups rouge : DescribeEngineDefaultClusterParameters rouge : DescribeEngineDefaultParameters rouge : DescribeEventCategories rouge : DescribeEventSubscriptions rouge : DescribeEvents rouge : DescribeExportTasks rouge : DescribeGlobalClusters rouge : DescribeIntegrations rouge : DescribeOptionGroupOptions rouge : DescribeOptionGroups rouge : DescribeOrderableDBInstanceOptions rouge : DescribePendingMaintenanceActions rouge : DescribeReservedDBInstances rouge : DescribeReservedDBInstancesOfferings rouge : DescribeSourceRegions rouge : DescribeTenantDatabases rouge : DescribeValidDBInstanceModifications rouge : DownloadCompleteDBLogFile rouge : DownloadDBLogFilePortion

Préfixe de service	Actions
	rds:FailoverDBCluster
	rouge : FailoverGlobalCluster
	rouge : ModifyActivityStream
	rouge : ModifyCertificates
	rouge : ModifyCurrentDBClusterCapacity
	rouge : ModifyDBClusterEndpoint
	rouge : ModifyDBClusterParameterGroup
	rouge : ModifyDBClusterSnapshotAttribute
	rouge : ModifyDBParameterGroup
	rds:ModifyDBProxy
	rouge : ModifyDBProxyEndpoint
	rouge : ModifyDBProxyTargetGroup
	rds:ModifyDBRecommendation
	rds:ModifyDBSnapshot
	rouge : ModifyDBSnapshotAttribute
	rouge : ModifyDBSubnetGroup
	rouge : ModifyEventSubscription
	rouge : ModifyGlobalCluster
	rouge : ModifyOptionGroup
	rouge : ModifyTenantDatabase
	rouge : PurchaseReservedDBInstancesOffering

Préfixe de service	Actions
	rds:RebootDBCluster
	rouge : RegisterDBProxyTargets
	rouge : RemoveFromGlobalCluster
	rouge : RemoveRoleFromDBCluster
	rouge : RemoveRoleFromDBInstance
	rouge : RemoveSourceIdentifierFromSubscription
	rouge : ResetDBClusterParameterGroup
	rouge : ResetDBParameterGroup
	rouge : RestoreDBClusterFromS3
	rouge : RestoreDBClusterFromSnapshot
	rouge : RestoreDBClusterToPointInTime
	rouge : RestoreDBInstanceFromDBSnapshot
	rouge : RestoreDBInstanceFromS3
	rouge : RestoreDBInstanceToPointInTime
	rouge : RevokeDBSecurityGroupIngress
	rouge : StartActivityStream
	rds:StartDBCluster
	rds:StartDBInstance
	rouge : StartDBInstanceAutomatedBackupsReplication
	rouge : StartExportTask
	rouge : StopActivityStream

Préfixe de service	Actions
	rds:StopDBCluster rds:StopDBInstance rouge : StopDBInstanceAutomatedBackupsReplication rouge : SwitchoverBlueGreenDeployment rouge : SwitchoverGlobalCluster rouge : SwitchoverReadReplica

Préfixe de service	Actions
redshift	décalage vers le rouge : AcceptReservedNodeExchange décalage vers le rouge : AddPartner décalage vers le rouge : AssociateDataShareConsumer décalage vers le rouge : AuthorizeClusterSecurityGroupIngress décalage vers le rouge : AuthorizeDataShare décalage vers le rouge : AuthorizeEndpointAccess décalage vers le rouge : AuthorizeSnapshotAccess décalage vers le rouge : BatchDeleteClusterSnapshots décalage vers le rouge : BatchModifyClusterSnapshots décalage vers le rouge : CancelResize décalage vers le rouge : CopyClusterSnapshot décalage vers le rouge : CreateAuthenticationProfile décalage vers le rouge : CreateCluster décalage vers le rouge : CreateClusterParameterGroup décalage vers le rouge : CreateClusterSecurityGroup décalage vers le rouge : CreateClusterSnapshot décalage vers le rouge : CreateClusterSubnetGroup décalage vers le rouge : CreateCustomDomainAssociation décalage vers le rouge : CreateEndpointAccess décalage vers le rouge : CreateEventSubscription décalage vers le rouge : CreateHsmClientCertificate

Préfixe de service	Actions
	décalage vers le rouge : CreateHsmConfiguration décalage vers le rouge : CreateIntegration décalage vers le rouge : CreateRedshiftIdcApplication décalage vers le rouge : CreateScheduledAction décalage vers le rouge : CreateSnapshotCopyGrant décalage vers le rouge : CreateSnapshotSchedule décalage vers le rouge : CreateUsageLimit décalage vers le rouge : DeauthorizeDataShare décalage vers le rouge : DeleteAuthenticationProfile décalage vers le rouge : DeleteCluster décalage vers le rouge : DeleteClusterParameterGroup décalage vers le rouge : DeleteClusterSecurityGroup décalage vers le rouge : DeleteClusterSnapshot décalage vers le rouge : DeleteClusterSubnetGroup décalage vers le rouge : DeleteCustomDomainAssociation décalage vers le rouge : DeleteEndpointAccess décalage vers le rouge : DeleteEventSubscription décalage vers le rouge : DeleteHsmClientCertificate décalage vers le rouge : DeleteHsmConfiguration décalage vers le rouge : DeletePartner décalage vers le rouge : DeleteRedshiftIdcApplication

Préfixe de service	Actions
	décalage vers le rouge : DeleteResourcePolicy décalage vers le rouge : DeleteScheduledAction décalage vers le rouge : DeleteSnapshotCopyGrant décalage vers le rouge : DeleteSnapshotSchedule décalage vers le rouge : DeleteUsageLimit décalage vers le rouge : DeregisterNamespace décalage vers le rouge : DescribeAccountAttributes décalage vers le rouge : DescribeAuthenticationProfiles décalage vers le rouge : DescribeClusterDbRevisions décalage vers le rouge : DescribeClusterParameterGroups décalage vers le rouge : DescribeClusterParameters décalage vers le rouge : DescribeClusterSecurityGroups décalage vers le rouge : DescribeClusterSnapshots décalage vers le rouge : DescribeClusterSubnetGroups décalage vers le rouge : DescribeClusterTracks décalage vers le rouge : DescribeClusterVersions décalage vers le rouge : DescribeClusters décalage vers le rouge : DescribeCustomDomainAssociations décalage vers le rouge : DescribeDataShares décalage vers le rouge : DescribeDataSharesForConsumer décalage vers le rouge : DescribeDataSharesForProducer

Préfixe de service	Actions
	décalage vers le rouge : DescribeDefaultClusterParameters
	décalage vers le rouge : DescribeEndpointAccess
	décalage vers le rouge : DescribeEndpointAuthorization
	décalage vers le rouge : DescribeEventCategories
	décalage vers le rouge : DescribeEventSubscriptions
	décalage vers le rouge : DescribeEvents
	décalage vers le rouge : DescribeHsmClientCertificates
	décalage vers le rouge : DescribeHsmConfigurations
	décalage vers le rouge : DescribeInboundIntegrations
	décalage vers le rouge : DescribeIntegrations
	décalage vers le rouge : DescribeLoggingStatus
	décalage vers le rouge : DescribeNodeConfigurationOptions
	décalage vers le rouge : DescribeOrderableClusterOptions
	décalage vers le rouge : DescribePartners
	décalage vers le rouge : DescribeRedshiftIdcApplications
	décalage vers le rouge : DescribeReservedNodeExchangeStatus
	décalage vers le rouge : DescribeReservedNodeOfferings
	décalage vers le rouge : DescribeReservedNodes
	décalage vers le rouge : DescribeResize
	décalage vers le rouge : DescribeScheduledActions
	décalage vers le rouge : DescribeSnapshotCopyGrants

Préfixe de service	Actions
	décalage vers le rouge : DescribeSnapshotSchedules décalage vers le rouge : DescribeStorage décalage vers le rouge : DescribeTableRestoreStatus décalage vers le rouge : DescribeUsageLimits décalage vers le rouge : DisableLogging décalage vers le rouge : DisableSnapshotCopy décalage vers le rouge : DisassociateDataShareConsumer décalage vers le rouge : EnableLogging décalage vers le rouge : EnableSnapshotCopy décalage vers le rouge : FailoverPrimaryCompute décalage vers le rouge : GetClusterCredentials décalage vers le rouge : GetClusterCredentialsWithIAM décalage vers le rouge : GetIdentityCenterAuthToken décalage vers le rouge : GetReservedNodeExchangeConfiguration Options décalage vers le rouge : GetReservedNodeExchangeOfferings décalage vers le rouge : GetResourcePolicy décalage vers le rouge : ListRecommendations décalage vers le rouge : ModifyAquaConfiguration décalage vers le rouge : ModifyAuthenticationProfile décalage vers le rouge : ModifyCluster

Préfixe de service	Actions
	décalage vers le rouge : ModifyClusterDbRevision décalage vers le rouge : ModifyClusterIamRoles décalage vers le rouge : ModifyClusterMaintenance décalage vers le rouge : ModifyClusterParameterGroup décalage vers le rouge : ModifyClusterSnapshot décalage vers le rouge : ModifyClusterSnapshotSchedule décalage vers le rouge : ModifyClusterSubnetGroup décalage vers le rouge : ModifyCustomDomainAssociation décalage vers le rouge : ModifyEndpointAccess décalage vers le rouge : ModifyEventSubscription décalage vers le rouge : ModifyRedshiftIdcApplication décalage vers le rouge : ModifyScheduledAction décalage vers le rouge : ModifySnapshotCopyRetentionPeriod décalage vers le rouge : ModifySnapshotSchedule décalage vers le rouge : ModifyUsageLimit décalage vers le rouge : PauseCluster décalage vers le rouge : PurchaseReservedNodeOffering décalage vers le rouge : PutResourcePolicy décalage vers le rouge : RebootCluster décalage vers le rouge : RegisterNamespace décalage vers le rouge : RejectDataShare

Préfixe de service	Actions
	décalage vers le rouge : ResetClusterParameterGroup
	décalage vers le rouge : ResizeCluster
	décalage vers le rouge : RestoreFromClusterSnapshot
	décalage vers le rouge : RestoreTableFromClusterSnapshot
	décalage vers le rouge : ResumeCluster
	décalage vers le rouge : RevokeClusterSecurityGroupIngress
	décalage vers le rouge : RevokeEndpointAccess
	décalage vers le rouge : RevokeSnapshotAccess
	décalage vers le rouge : RotateEncryptionKey
	décalage vers le rouge : UpdatePartnerStatus
redshift-data	données redshift : BatchExecuteStatement
	données redshift : CancelStatement
	données redshift : DescribeStatement
	données redshift : DescribeTable
	données redshift : ExecuteStatement
	données redshift : GetStatementResult
	données redshift : ListDatabases
	données redshift : ListSchemas
	données redshift : ListSessions
	données redshift : ListStatements
	données redshift : ListTables

Préfixe de service	Actions
refactor-spaces	espaces de refactorisation : CreateApplication espaces de refactorisation : CreateEnvironment espaces de refactorisation : CreateRoute espaces de refactorisation : CreateService espaces de refactorisation : DeleteApplication espaces de refactorisation : DeleteEnvironment espaces de refactorisation : DeleteResourcePolicy espaces de refactorisation : DeleteRoute espaces de refactorisation : DeleteService espaces de refactorisation : GetApplication espaces de refactorisation : GetEnvironment espaces de refactorisation : GetResourcePolicy espaces de refactorisation : GetRoute espaces de refactorisation : GetService espaces de refactorisation : ListApplications espaces de refactorisation : ListEnvironmentVpcs espaces de refactorisation : ListEnvironments espaces de refactorisation : ListRoutes espaces de refactorisation : ListServices espaces de refactorisation : PutResourcePolicy espaces de refactorisation : UpdateRoute

Préfixe de service	Actions
rekognition	reconnaissance : AssociateFaces
	reconnaissance : CompareFaces
	reconnaissance : CopyProjectVersion
	reconnaissance : CreateCollection
	reconnaissance : CreateDataset
	reconnaissance : CreateFaceLivenessSession
	reconnaissance : CreateProject
	reconnaissance : CreateProjectVersion
	reconnaissance : CreateStreamProcessor
	reconnaissance : CreateUser
	reconnaissance : DeleteCollection
	reconnaissance : DeleteDataset
	reconnaissance : DeleteFaces
	reconnaissance : DeleteProject
	reconnaissance : DeleteProjectPolicy
	reconnaissance : DeleteProjectVersion
	reconnaissance : DeleteStreamProcessor
	reconnaissance : DeleteUser
	reconnaissance : DescribeCollection
	reconnaissance : DescribeDataset
	reconnaissance : DescribeProjectVersions

Préfixe de service	Actions
	reconnaissance : DescribeProjects
	reconnaissance : DescribeStreamProcessor
	reconnaissance : DetectCustomLabels
	reconnaissance : DetectFaces
	reconnaissance : DetectLabels
	reconnaissance : DetectModerationLabels
	reconnaissance : DetectProtectiveEquipment
	reconnaissance : DetectText
	reconnaissance : DisassociateFaces
	reconnaissance : DistributeDatasetEntries
	reconnaissance : GetCelebrityInfo
	reconnaissance : GetCelebrityRecognition
	reconnaissance : GetContentModeration
	reconnaissance : GetFaceDetection
	reconnaissance : GetFaceLivenessSessionResults
	reconnaissance : GetFaceSearch
	reconnaissance : GetLabelDetection
	reconnaissance : GetMediaAnalysisJob
	reconnaissance : GetPersonTracking
	reconnaissance : GetSegmentDetection
	reconnaissance : GetTextDetection

Préfixe de service	Actions
	reconnaissance : IndexFaces
	reconnaissance : ListCollections
	reconnaissance : ListDatasetEntries
	reconnaissance : ListDatasetLabels
	reconnaissance : ListFaces
	reconnaissance : ListMediaAnalysisJobs
	reconnaissance : ListProjectPolicies
	reconnaissance : ListStreamProcessors
	reconnaissance : ListUsers
	reconnaissance : PutProjectPolicy
	reconnaissance : RecognizeCelebrities
	reconnaissance : SearchFaces
	reconnaissance : SearchFacesByImage
	reconnaissance : SearchUsers
	reconnaissance : SearchUsersByImage
	reconnaissance : StartCelebrityRecognition
	reconnaissance : StartContentModeration
	reconnaissance : StartFaceDetection
	reconnaissance : StartFaceLivenessSession
	reconnaissance : StartFaceSearch
	reconnaissance : StartLabelDetection

Préfixe de service	Actions
	reconnaissance : StartMediaAnalysisJob
	reconnaissance : StartPersonTracking
	reconnaissance : StartProjectVersion
	reconnaissance : StartSegmentDetection
	reconnaissance : StartStreamProcessor
	reconnaissance : StartTextDetection
	reconnaissance : StopProjectVersion
	reconnaissance : StopStreamProcessor
	reconnaissance : UpdateDatasetEntries
	reconnaissance : UpdateStreamProcessor

Préfixe de service	Actions
resiliencehub	hub de résilience : AcceptResourceGroupingRecommendations hub de résilience : AddDraftAppVersionResourceMappings hub de résilience : BatchUpdateRecommendationStatus hub de résilience : CreateApp hub de résilience : CreateAppVersionAppComponent hub de résilience : CreateAppVersionResource hub de résilience : CreateRecommendationTemplate hub de résilience : CreateResiliencyPolicy hub de résilience : DeleteApp hub de résilience : DeleteAppAssessment hub de résilience : DeleteAppInputSource hub de résilience : DeleteAppVersionAppComponent hub de résilience : DeleteAppVersionResource hub de résilience : DeleteRecommendationTemplate hub de résilience : DeleteResiliencyPolicy hub de résilience : DescribeApp hub de résilience : DescribeAppAssessment hub de résilience : DescribeAppVersion hub de résilience : DescribeAppVersionAppComponent hub de résilience : DescribeAppVersionResource hub de résilience : DescribeAppVersionResourcesResolutionStatus

Préfixe de service	Actions
	hub de résilience : DescribeAppVersionTemplate hub de résilience : DescribeDraftAppVersionResourcesImportStatus hub de résilience : DescribeMetricsExport hub de résilience : DescribeResiliencyPolicy hub de résilience : DescribeResourceGroupingRecommendationTask hub de résilience : ImportResourcesToDraftAppVersion hub de résilience : ListAlarmRecommendations hub de résilience : ListAppAssessmentComplianceDrifts hub de résilience : ListAppAssessmentResourceDrifts hub de résilience : ListAppAssessments hub de résilience : ListAppComponentCompliances hub de résilience : ListAppComponentRecommendations hub de résilience : ListAppInputSources hub de résilience : ListAppVersionAppComponents hub de résilience : ListAppVersionResourceMappings hub de résilience : ListAppVersionResources hub de résilience : ListAppVersions hub de résilience : ListApps hub de résilience : ListMetrics hub de résilience : ListRecommendationTemplates

Préfixe de service	Actions
	hub de résilience : ListResiliencyPolicies hub de résilience : ListResourceGroupingRecommendations hub de résilience : ListSopRecommendations hub de résilience : ListSuggestedResiliencyPolicies hub de résilience : ListTestRecommendations hub de résilience : ListUnsupportedAppVersionResources hub de résilience : PublishAppVersion hub de résilience : PutDraftAppVersionTemplate hub de résilience : RejectResourceGroupingRecommendations hub de résilience : RemoveDraftAppVersionResourceMappings hub de résilience : ResolveAppVersionResources hub de résilience : StartAppAssessment hub de résilience : StartResourceGroupingRecommendationTask hub de résilience : UpdateApp hub de résilience : UpdateAppVersion hub de résilience : UpdateAppVersionAppComponent hub de résilience : UpdateAppVersionResource hub de résilience : UpdateResiliencyPolicy

Préfixe de service	Actions
resource-explorer-2	explorateur de ressources-2 : AssociateDefaultView
	explorateur de ressources-2 : BatchGetView
	explorateur de ressources-2 : CreateIndex
	explorateur de ressources-2 : CreateResourceExplorerSetup
	explorateur de ressources-2 : CreateView
	explorateur de ressources-2 : DeleteIndex
	explorateur de ressources-2 : DeleteResourceExplorerSetup
	explorateur de ressources-2 : DeleteView
	explorateur de ressources-2 : DisassociateDefaultView
	explorateur de ressources-2 : GetAccountLevelServiceConfiguration
	explorateur de ressources-2 : GetDefaultView
	explorateur de ressources-2 : GetIndex
	explorateur de ressources-2 : GetManagedView
	explorateur de ressources-2 : GetResourceExplorerSetup
	explorateur de ressources-2 : GetServiceIndex
	explorateur de ressources-2 : GetServiceView
	explorateur de ressources-2 : ListIndexes
	explorateur de ressources-2 : ListIndexesForMembers
	explorateur de ressources-2 : ListManagedViews
	explorateur de ressources-2 : ListServiceIndexes
	explorateur de ressources-2 : ListServiceViews

Préfixe de service	Actions
	explorateur de ressources-2 : ListStreamingAccessForServices
	explorateur de ressources-2 : ListSupportedResourceTypes
	explorateur de ressources-2 : ListViews
	resource-explorer-2:Search
	explorateur de ressources-2 : UpdateIndexType
	explorateur de ressources-2 : UpdateView

Préfixe de service	Actions
resource-groups	groupes de ressources : CancelTagSyncTask groupes de ressources : GetAccountSettings groupes de ressources : GetGroup groupes de ressources : GetGroupConfiguration groupes de ressources : GetGroupQuery groupes de ressources : GetTagSyncTask groupes de ressources : GroupResources groupes de ressources : ListGroupResources groupes de ressources : ListGroupingStatuses groupes de ressources : ListGroups groupes de ressources : ListTagSyncTasks groupes de ressources : PutGroupConfiguration groupes de ressources : SearchResources groupes de ressources : StartTagSyncTask groupes de ressources : UngroupResources groupes de ressources : UpdateAccountSettings groupes de ressources : UpdateGroup groupes de ressources : UpdateGroupQuery

Préfixe de service	Actions
robomaker	fabricant de robots : BatchDeleteWorlds
	fabricant de robots : BatchDescribeSimulationJob
	fabricant de robots : CancelDeploymentJob
	fabricant de robots : CancelSimulationJob
	fabricant de robots : CancelSimulationJobBatch
	fabricant de robots : CancelWorldExportJob
	fabricant de robots : CancelWorldGenerationJob
	fabricant de robots : CreateDeploymentJob
	fabricant de robots : CreateFleet
	fabricant de robots : CreateRobot
	fabricant de robots : CreateRobotApplication
	fabricant de robots : CreateRobotApplicationVersion
	fabricant de robots : CreateSimulationApplication
	fabricant de robots : CreateSimulationApplicationVersion
	fabricant de robots : CreateSimulationJob
	fabricant de robots : CreateWorldExportJob
	fabricant de robots : CreateWorldGenerationJob
	fabricant de robots : CreateWorldTemplate
	fabricant de robots : DeleteFleet
	fabricant de robots : DeleteRobot
	fabricant de robots : DeleteRobotApplication

Préfixe de service	Actions
	fabricant de robots : DeleteSimulationApplication
	fabricant de robots : DeleteWorldTemplate
	fabricant de robots : DeregisterRobot
	fabricant de robots : DescribeDeploymentJob
	fabricant de robots : DescribeFleet
	fabricant de robots : DescribeRobot
	fabricant de robots : DescribeRobotApplication
	fabricant de robots : DescribeSimulationApplication
	fabricant de robots : DescribeSimulationJob
	fabricant de robots : DescribeSimulationJobBatch
	fabricant de robots : DescribeWorld
	fabricant de robots : DescribeWorldExportJob
	fabricant de robots : DescribeWorldGenerationJob
	fabricant de robots : DescribeWorldTemplate
	fabricant de robots : GetWorldTemplateBody
	fabricant de robots : ListDeploymentJobs
	fabricant de robots : ListFleets
	fabricant de robots : ListRobotApplications
	fabricant de robots : ListRobots
	fabricant de robots : ListSimulationApplications
	fabricant de robots : ListSimulationJobBatches

Préfixe de service	Actions
	fabricant de robots : ListSimulationJobs
	fabricant de robots : ListWorldExportJobs
	fabricant de robots : ListWorldGenerationJobs
	fabricant de robots : ListWorldTemplates
	fabricant de robots : ListWorlds
	fabricant de robots : RegisterRobot
	fabricant de robots : RestartSimulationJob
	fabricant de robots : StartSimulationJobBatch
	fabricant de robots : SyncDeploymentJob
	fabricant de robots : UpdateRobotApplication
	fabricant de robots : UpdateSimulationApplication
	fabricant de robots : UpdateWorldTemplate

Préfixe de service	Actions
rolesanywhere	rôles n'importe où : CreateProfile rôles n'importe où : CreateTrustAnchor rôles n'importe où : DeleteAttributeMapping rôles n'importe où : DeleteCrl rôles n'importe où : DeleteProfile rôles n'importe où : DeleteTrustAnchor rôles n'importe où : DisableCrl rôles n'importe où : DisableProfile rôles n'importe où : DisableTrustAnchor rôles n'importe où : EnableCrl rôles n'importe où : EnableProfile rôles n'importe où : EnableTrustAnchor rôles n'importe où : GetCrl rôles n'importe où : GetProfile rôles n'importe où : GetSubject rôles n'importe où : GetTrustAnchor rôles n'importe où : ImportCrl rôles n'importe où : ListCrls rôles n'importe où : ListProfiles rôles n'importe où : ListSubjects rôles n'importe où : ListTrustAnchors

Préfixe de service	Actions
	rôles n'importe où : PutAttributeMapping rôles n'importe où : PutNotificationSettings rôles n'importe où : ResetNotificationSettings rôles n'importe où : UpdateCrl rôles n'importe où : UpdateProfile rôles n'importe où : UpdateTrustAnchor

Préfixe de service	Actions
route53	route 53 : ActivateKeySigningKey
	route 53 : AssociateVPCWithHostedZone
	route 53 : ChangeCidrCollection
	route 53 : ChangeResourceRecordSets
	route 53 : CreateCidrCollection
	route 53 : CreateHealthCheck
	route 53 : CreateHostedZone
	route 53 : CreateKeySigningKey
	route 53 : CreateQueryLoggingConfig
	route 53 : CreateReusableDelegationSet
	route 53 : CreateTrafficPolicy
	route 53 : CreateTrafficPolicyInstance
	route 53 : CreateTrafficPolicyVersion
	route 53 : CreateVPCAssociationAuthorization
	route 53 : DeactivateKeySigningKey
	route 53 : DeleteCidrCollection
	route 53 : DeleteHealthCheck
	route 53 : DeleteHostedZone
	route 53 : DeleteKeySigningKey
	route 53 : DeleteQueryLoggingConfig
	route 53 : DeleteReusableDelegationSet

Préfixe de service	Actions
	route 53 : DeleteTrafficPolicy
	route 53 : DeleteTrafficPolicyInstance
	route 53 : DeleteVPCAssociationAuthorization
	route 53 : DisableHostedZoneDNSSEC
	route 53 : DisassociateVPCFromHostedZone
	route 53 : EnableHostedZoneDNSSEC
	route 53 : GetAccountLimit
	route 53 : GetChange
	route 53 : GetCheckerIpRanges
	route53:GetDNSSEC
	route 53 : GetGeoLocation
	route 53 : GetHealthCheck
	route 53 : GetHealthCheckCount
	route 53 : GetHealthCheckLastFailureReason
	route 53 : GetHealthCheckStatus
	route 53 : GetHostedZone
	route 53 : GetHostedZoneCount
	route 53 : GetHostedZoneLimit
	route 53 : GetQueryLoggingConfig
	route 53 : GetReusableDelegationSet
	route 53 : GetReusableDelegationSetLimit

Préfixe de service	Actions
	route 53 : GetTrafficPolicy
	route 53 : GetTrafficPolicyInstance
	route 53 : GetTrafficPolicyInstanceCount
	route 53 : ListCidrBlocks
	route 53 : ListCidrCollections
	route 53 : ListCidrLocations
	route 53 : ListGeoLocations
	route 53 : ListHealthChecks
	route 53 : ListHostedZones
	route 53 : ListHostedZonesByName
	route 53 : ListHostedZonesByVPC
	route 53 : ListQueryLoggingConfigs
	route 53 : ListResourceRecordSets
	route 53 : ListReusableDelegationSets
	route 53 : ListTrafficPolicies
	route 53 : ListTrafficPolicyInstances
	route 53 : ListTrafficPolicyInstancesByHostedZone
	route 53 : ListTrafficPolicyInstancesByPolicy
	route 53 : ListTrafficPolicyVersions
	route 53 : ListVPCAssociationAuthorizations
	route53:TestDNSAnswer

Préfixe de service	Actions
	route 53 : UpdateHealthCheck route 53 : UpdateHostedZoneComment route 53 : UpdateTrafficPolicyComment route 53 : UpdateTrafficPolicyInstance

Préfixe de service	Actions
route53-recovery-control-config	route53-recovery-control-config : CreateCluster route53-recovery-control-config : CreateControlPanel route53-recovery-control-config : CreateRoutingControl route53-recovery-control-config : CreateSafetyRule route53-recovery-control-config : DeleteCluster route53-recovery-control-config : DeleteControlPanel route53-recovery-control-config : DeleteRoutingControl route53-recovery-control-config : DeleteSafetyRule route53-recovery-control-config : DescribeCluster route53-recovery-control-config : DescribeControlPanel route53-recovery-control-config : DescribeRoutingControl route53-recovery-control-config : DescribeSafetyRule route53-recovery-control-config : GetResourcePolicy route53-recovery-control-config : ListAssociatedRoute53HealthChecks route53-recovery-control-config : ListClusters route53-recovery-control-config : ListControlPanels route53-recovery-control-config : ListRoutingControls route53-recovery-control-config : ListSafetyRules route53-recovery-control-config : UpdateCluster route53-recovery-control-config : UpdateControlPanel

Préfixe de service	Actions
	route53-recovery-control-config : UpdateRoutingControl route53-recovery-control-config : UpdateSafetyRule

Préfixe de service	Actions
route53-recovery-readiness	route 53 - préparation au rétablissement : CreateCell
	route 53 - préparation au rétablissement : CreateCrossAccount Authorization
	route 53 - préparation au rétablissement : CreateReadinessCheck
	route 53 - préparation au rétablissement : CreateRecoveryGroup
	route 53 - préparation au rétablissement : CreateResourceSet
	route 53 - préparation au rétablissement : DeleteCell
	route 53 - préparation au rétablissement : DeleteCrossAccount Authorization
	route 53 - préparation au rétablissement : DeleteReadinessCheck
	route 53 - préparation au rétablissement : DeleteRecoveryGroup
	route 53 - préparation au rétablissement : DeleteResourceSet
	route 53 - préparation au rétablissement : GetArchitectureRecommendations
	route 53 - préparation au rétablissement : GetCell
	route 53 - préparation au rétablissement : GetCellReadinessSummary
	route 53 - préparation au rétablissement : GetReadinessCheck
	route 53 - préparation au rétablissement : GetReadinessCheckResourceStatus
	route 53 - préparation au rétablissement : GetReadinessCheckStatus
	route 53 - préparation au rétablissement : GetRecoveryGroup

Préfixe de service	Actions
	route 53 - préparation au rétablissement : GetRecoveryGroupReadinessSummary
	route 53 - préparation au rétablissement : GetResourceSet
	route 53 - préparation au rétablissement : ListCells
	route 53 - préparation au rétablissement : ListCrossAccountAuthorizations
	route 53 - préparation au rétablissement : ListReadinessChecks
	route 53 - préparation au rétablissement : ListRecoveryGroups
	route 53 - préparation au rétablissement : ListResourceSets
	route 53 - préparation au rétablissement : ListRules
	route 53 - préparation au rétablissement : UpdateCell
	route 53 - préparation au rétablissement : UpdateReadinessCheck
	route 53 - préparation au rétablissement : UpdateRecoveryGroup
	route 53 - préparation au rétablissement : UpdateResourceSet

Préfixe de service	Actions
route53resolver	résolveur Route53 : AssociateFirewallRuleGroup résolveur Route53 : AssociateResolverEndpointIpAddress résolveur Route53 : AssociateResolverQueryLogConfig résolveur Route53 : AssociateResolverRule résolveur Route53 : CreateFirewallDomainList résolveur Route53 : CreateFirewallRule résolveur Route53 : CreateFirewallRuleGroup résolveur Route53 : CreateResolverEndpoint résolveur Route53 : CreateResolverQueryLogConfig résolveur Route53 : CreateResolverRule résolveur Route53 : DeleteFirewallDomainList résolveur Route53 : DeleteFirewallRule résolveur Route53 : DeleteFirewallRuleGroup résolveur Route53 : DeleteOutpostResolver résolveur Route53 : DeleteResolverEndpoint résolveur Route53 : DeleteResolverQueryLogConfig résolveur Route53 : DeleteResolverRule résolveur Route53 : DisassociateFirewallRuleGroup résolveur Route53 : DisassociateResolverEndpointIpAddress résolveur Route53 : DisassociateResolverQueryLogConfig résolveur Route53 : DisassociateResolverRule

Préfixe de service	Actions
	résolveur Route53 : GetFirewallConfig résolveur Route53 : GetFirewallDomainList résolveur Route53 : GetFirewallRuleGroup résolveur Route53 : GetFirewallRuleGroupAssociation résolveur Route53 : GetFirewallRuleGroupPolicy résolveur Route53 : GetOutpostResolver résolveur Route53 : GetResolverConfig résolveur Route53 : GetResolverDnssecConfig résolveur Route53 : GetResolverEndpoint résolveur Route53 : GetResolverQueryLogConfig résolveur Route53 : GetResolverQueryLogConfigAssociation résolveur Route53 : GetResolverQueryLogConfigPolicy résolveur Route53 : GetResolverRule résolveur Route53 : GetResolverRuleAssociation résolveur Route53 : GetResolverRulePolicy résolveur Route53 : ImportFirewallDomains résolveur Route53 : ListFirewallConfigs résolveur Route53 : ListFirewallDomainLists résolveur Route53 : ListFirewallDomains résolveur Route53 : ListFirewallRuleGroupAssociations résolveur Route53 : ListFirewallRuleGroups

Préfixe de service	Actions
	résolveur Route53 : ListFirewallRuleTypes résolveur Route53 : ListFirewallRules résolveur Route53 : ListOutpostResolvers résolveur Route53 : ListResolverConfigs résolveur Route53 : ListResolverDnssecConfigs résolveur Route53 : ListResolverEndpointIpAddresses résolveur Route53 : ListResolverEndpoints résolveur Route53 : ListResolverQueryLogConfigAssociations résolveur Route53 : ListResolverQueryLogConfigs résolveur Route53 : ListResolverRuleAssociations résolveur Route53 : ListResolverRules résolveur Route53 : PutFirewallRuleGroupPolicy résolveur Route53 : PutResolverQueryLogConfigPolicy résolveur Route53 : UpdateFirewallConfig résolveur Route53 : UpdateFirewallDomains résolveur Route53 : UpdateFirewallRule résolveur Route53 : UpdateFirewallRuleGroupAssociation résolveur Route53 : UpdateOutpostResolver résolveur Route53 : UpdateResolverConfig résolveur Route53 : UpdateResolverDnssecConfig résolveur Route53 : UpdateResolverEndpoint

Préfixe de service	Actions
	résolveur Route53 : UpdateResolverRule
rum	rhum : BatchCreateRumMetricDefinitions
	rhum : BatchDeleteRumMetricDefinitions
	rhum : BatchGetRumMetricDefinitions
	rhum : CreateAppMonitor
	rhum : DeleteAppMonitor
	rhum : DeleteResourcePolicy
	rhum : DeleteRumMetricsDestination
	rhum : GetAppMonitor
	rhum : GetAppMonitorData
	rhum : GetResourcePolicy
	rhum : ListAppMonitors
	rhum : ListRumMetricsDestinations
	rhum : PutResourcePolicy
	rhum : PutRumMetricsDestination
	rhum : UpdateAppMonitor
	rhum : UpdateRumMetricDefinition

Préfixe de service	Actions
s3	s3 : AssociateAccessGrantsIdentityCenter s3 : CreateAccessGrant s3 : CreateAccessGrantsInstance s3 : CreateAccessGrantsLocation s3 : CreateAccessPoint s3 : CreateAccessPointForObjectLambda s3 : CreateBucket s3 : CreateBucketMetadataTableConfiguration s3 : CreateJob s3 : CreateMultiRegionAccessPoint s3 : DeleteAccessGrant s3 : DeleteAccessGrantsInstance s3 : DeleteAccessGrantsInstanceResourcePolicy s3 : DeleteAccessGrantsLocation s3 : DeleteAccessPoint s3 : DeleteAccessPointForObjectLambda s3 : DeleteAccessPointPolicy s3 : DeleteAccessPointPolicyForObjectLambda s3 : DeleteBucket s3 : DeleteBucketMetadataTableConfiguration s3 : DeleteBucketPolicy

Préfixe de service	Actions
	s3 : DeleteBucketWebsite s3 : DeleteMultiRegionAccessPoint s3 : DeleteStorageLensConfiguration s3 : DescribeJob s3 : DescribeMultiRegionAccessPointOperation s3 : DissociateAccessGrantsIdentityCenter s3 : GetAccelerateConfiguration s3 : GetAccessGrant s3 : GetAccessGrantsInstance s3 : GetAccessGrantsInstanceForPrefix s3 : GetAccessGrantsInstanceResourcePolicy s3 : GetAccessGrantsLocation s3 : GetAccessPoint s3 : GetAccessPointConfigurationForObjectLambda s3 : GetAccessPointForObjectLambda s3 : GetAccessPointPolicy s3 : GetAccessPointPolicyForObjectLambda s3 : GetAccessPointPolicyStatus s3 : GetAccessPointPolicyStatusForObjectLambda s3 : GetAccountPublicAccessBlock s3 : GetAnalyticsConfiguration

Préfixe de service	Actions
	s3 : GetBucketAbac s3 : GetBucketAcl s3 : GetBucket CORS s3 : GetBucketLocation s3 : GetBucketLogging s3 : GetBucketNotification s3 : GetBucketObjectLockConfiguration s3 : GetBucketOwnershipControls s3 : GetBucketPolicy s3 : GetBucketPolicyStatus s3 : GetBucketPublicAccessBlock s3 : GetBucketRequestPayment s3 : GetBucketVersioning s3 : GetBucketWebsite s3 : GetDataAccess s3 : GetEncryptionConfiguration s3 : GetIntelligentTieringConfiguration s3 : GetInventoryConfiguration s3 : GetLifecycleConfiguration s3 : GetMetricsConfiguration s3 : GetMultiRegionAccessPoint

Préfixe de service	Actions
	s3 : GetMultiRegionAccessPointPolicy s3 : GetMultiRegionAccessPointPolicyStatus s3 : GetMultiRegionAccessPointRoutes s3 : GetReplicationConfiguration s3 : GetStorageLensConfiguration s3 : GetStorageLensDashboard s3 : ListAccessGrants s3 : ListAccessGrantsInstances s3 : ListAccessGrantsLocations s3 : ListAccessPoints s3 : ListAccessPointsForObjectLambda s3 : ListAllMyBuckets s3 : ListBucketMultipartUploads s3 : ListCallerAccessGrants s3 : ListJobs s3 : ListMultiRegionAccessPoints s3 : ListStorageLensConfigurations s3 : PutAccelerateConfiguration s3 : PutAccessGrantsInstanceResourcePolicy s3 : PutAccessPointConfigurationForObjectLambda s3 : PutAccessPointPolicy

Préfixe de service	Actions
	s3 : PutAccessPointPolicyForObjectLambda s3 : PutAccountPublicAccessBlock s3 : PutAnalyticsConfiguration s3 : PutBucketAbac s3 : PutBucketAcl s3 : PutBucket CORS s3 : PutBucketLogging s3 : PutBucketNotification s3 : PutBucketObjectLockConfiguration s3 : PutBucketOwnershipControls s3 : PutBucketPolicy s3 : PutBucketPublicAccessBlock s3 : PutBucketRequestPayment s3 : PutBucketVersioning s3 : PutBucketWebsite s3 : PutEncryptionConfiguration s3 : PutIntelligentTieringConfiguration s3 : PutInventoryConfiguration s3 : PutLifecycleConfiguration s3 : PutMetricsConfiguration s3 : PutMultiRegionAccessPointPolicy

Préfixe de service	Actions
	s3 : PutReplicationConfiguration s3 : PutStorageLensConfiguration s3 : SubmitMultiRegionAccessPointRoutes s3 : UpdateAccessGrantsLocation s3 : UpdateBucketMetadataAnnotationTableConfiguration s3 : UpdateBucketMetadataJournalTableConfiguration s3 : UpdateJobPriority s3 : UpdateJobStatus
s3-outposts	avant-postes S3 : CreateEndpoint avant-postes S3 : DeleteEndpoint avant-postes S3 : ListEndpoints avant-postes S3 : ListOutpostsWithS3 avant-postes S3 : ListSharedEndpoints

Préfixe de service	Actions
sagemaker-geospatial	sagemaker-geospatial : DeleteEarthObservationJob sagemaker-geospatial : DeleteVectorEnrichmentJob sagemaker-geospatial : ExportEarthObservationJob sagemaker-geospatial : ExportVectorEnrichmentJob sagemaker-geospatial : GetEarthObservationJob sagemaker-geospatial : GetRasterDataCollection sagemaker-geospatial : GetTile sagemaker-geospatial : GetVectorEnrichmentJob sagemaker-geospatial : ListEarthObservationJobs sagemaker-geospatial : ListRasterDataCollections sagemaker-geospatial : ListVectorEnrichmentJobs sagemaker-geospatial : SearchRasterDataCollection sagemaker-geospatial : StartEarthObservationJob sagemaker-geospatial : StartVectorEnrichmentJob sagemaker-geospatial : StopEarthObservationJob sagemaker-geospatial : StopVectorEnrichmentJob

Préfixe de service	Actions
savingsplans	plans d'épargne : CreateSavingsPlan
	plans d'épargne : DeleteQueuedSavingsPlan
	plans d'épargne : DescribeSavingsPlanRates
	plans d'épargne : DescribeSavingsPlans
	plans d'épargne : DescribeSavingsPlansOfferingRates
	plans d'épargne : DescribeSavingsPlansOfferings
	plans d'épargne : ReturnSavingsPlan

Préfixe de service	Actions
schemas	schémas : CreateDiscoverer schémas : CreateRegistry schémas : CreateSchema schémas : DeleteDiscoverer schémas : DeleteRegistry schémas : DeleteResourcePolicy schémas : DeleteSchema schémas : DeleteSchemaVersion schémas : DescribeCodeBinding schémas : DescribeDiscoverer schémas : DescribeRegistry schémas : DescribeSchema schémas : ExportSchema schémas : GetCodeBindingSource schémas : GetDiscoveredSchema schémas : GetResourcePolicy schémas : ListDiscoverers schémas : ListRegistries schémas : ListSchemaVersions schémas : ListSchemas schémas : PutCodeBinding

Préfixe de service	Actions
	schémas : PutResourcePolicy schémas : SearchSchemas schémas : StartDiscoverer schémas : StopDiscoverer schémas : UpdateDiscoverer schémas : UpdateRegistry schémas : UpdateSchema
sdb	sdb : CreateDomain sdb : DeleteDomain sdb : DomainMetadata sdb : ListDomains

Préfixe de service	Actions
secretsmanager	responsable des secrets : CancelRotateSecret responsable des secrets : CreateSecret responsable des secrets : DeleteResourcePolicy responsable des secrets : DeleteSecret responsable des secrets : DescribeSecret responsable des secrets : GetRandomPassword responsable des secrets : GetResourcePolicy responsable des secrets : GetSecretValue responsable des secrets : ListSecretVersionIds responsable des secrets : ListSecrets responsable des secrets : PutResourcePolicy responsable des secrets : PutSecretValue responsable des secrets : RemoveRegionsFromReplication responsable des secrets : ReplicateSecretToRegions responsable des secrets : RestoreSecret responsable des secrets : RotateSecret responsable des secrets : StopReplicationToReplica responsable des secrets : UpdateSecret responsable des secrets : ValidateResourcePolicy

Préfixe de service	Actions
securityhub	hub de sécurité : AcceptAdministratorInvitation
	hub de sécurité : AcceptInvitation
	hub de sécurité : BatchDeleteAutomationRules
	hub de sécurité : BatchDisableStandards
	hub de sécurité : BatchEnableStandards
	hub de sécurité : BatchGetAutomationRules
	hub de sécurité : BatchGetConfigurationPolicyAssociations
	hub de sécurité : BatchGetEnabledRegionsV2
	hub de sécurité : BatchGetSecurityControls
	hub de sécurité : BatchGetStandardsControlAssociations
	hub de sécurité : BatchImportFindings
	hub de sécurité : BatchUpdateAutomationRules
	hub de sécurité : BatchUpdateFindings
	hub de sécurité : BatchUpdateStandardsControlAssociations
	hub de sécurité : V2 ConnectorRegistrations
	hub de sécurité : CreateActionTarget
	hub de sécurité : V2 CreateAggregator
	hub de sécurité : CreateAutomationRule
	hub de sécurité : CreateAutomationRuleV2
	hub de sécurité : CreateConfigurationPolicy
	hub de sécurité : CreateConnector

Préfixe de service	Actions
	hub de sécurité : V2 CreateConnector
	hub de sécurité : CreateFindingAggregator
	hub de sécurité : CreateInsight
	hub de sécurité : CreateMembers
	hub de sécurité : V2 CreateTicket
	hub de sécurité : DeclineInvitations
	hub de sécurité : DeleteActionTarget
	hub de sécurité : V2 DeleteAggregator
	hub de sécurité : DeleteAutomationRuleV2
	hub de sécurité : DeleteConfigurationPolicy
	hub de sécurité : DeleteConnector
	hub de sécurité : V2 DeleteConnector
	hub de sécurité : DeleteFindingAggregator
	hub de sécurité : DeleteInsight
	hub de sécurité : DeleteInvitations
	hub de sécurité : DeleteMembers
	hub de sécurité : DescribeActionTargets
	hub de sécurité : DescribeHub
	hub de sécurité : DescribeOrganizationConfiguration
	hub de sécurité : DescribeProducts
	hub de sécurité : DescribeSecurityHubV2

Préfixe de service	Actions
	hub de sécurité : DescribeStandards
	hub de sécurité : DisableImportFindingsForProduct
	hub de sécurité : DisableOrganizationAdminAccount
	hub de sécurité : DisableSecurityHub
	hub de sécurité : DisableSecurityHubFeatureV2
	hub de sécurité : DisableSecurityHubV2
	hub de sécurité : DisassociateFromAdministratorAccount
	hub de sécurité : DisassociateFromMasterAccount
	hub de sécurité : DisassociateMembers
	hub de sécurité : EnableImportFindingsForProduct
	hub de sécurité : EnableOrganizationAdminAccount
	hub de sécurité : EnableSecurityHub
	hub de sécurité : EnableSecurityHubFeatureV2
	hub de sécurité : GetAdministratorAccount
	hub de sécurité : V2 GetAggregator
	hub de sécurité : GetAutomationRuleV2
	hub de sécurité : GetConfigurationPolicy
	hub de sécurité : GetConfigurationPolicyAssociation
	hub de sécurité : GetConnector
	hub de sécurité : V2 GetConnector
	hub de sécurité : GetCoverageStatisticsV2

Préfixe de service	Actions
	hub de sécurité : GetEnabledStandards
	hub de sécurité : GetFindingAggregator
	hub de sécurité : GetFindingHistory
	hub de sécurité : GetFindings
	hub de sécurité : GetInsightResults
	hub de sécurité : GetInsights
	hub de sécurité : GetInvitationsCount
	hub de sécurité : GetMasterAccount
	hub de sécurité : GetMembers
	hub de sécurité : GetSecurityControlDefinition
	hub de sécurité : InviteMembers
	hub de sécurité : V2 ListAggregators
	hub de sécurité : ListAutomationRules
	hub de sécurité : ListAutomationRulesV2
	hub de sécurité : ListConfigurationPolicies
	hub de sécurité : ListConfigurationPolicyAssociations
	hub de sécurité : ListConnectors
	hub de sécurité : V2 ListConnectors
	hub de sécurité : ListEnabledProductsForImport
	hub de sécurité : ListFindingAggregators
	hub de sécurité : ListInvitations

Préfixe de service	Actions
	hub de sécurité : ListMembers
	hub de sécurité : ListOrganizationAdminAccounts
	hub de sécurité : ListSecurityControlDefinitions
	hub de sécurité : ListStandardsControlAssociations
	hub de sécurité : StartConfigurationPolicyAssociation
	hub de sécurité : StartConfigurationPolicyDisassociation
	hub de sécurité : UpdateActionTarget
	hub de sécurité : V2 UpdateAggregator
	hub de sécurité : UpdateAutomationRuleV2
	hub de sécurité : UpdateConfigurationPolicy
	hub de sécurité : UpdateConnector
	hub de sécurité : V2 UpdateConnector
	hub de sécurité : UpdateFindingAggregator
	hub de sécurité : UpdateFindings
	hub de sécurité : UpdateInsight
	hub de sécurité : UpdateOrganizationConfiguration
	hub de sécurité : UpdateSecurityControl
	hub de sécurité : UpdateSecurityHubConfiguration

Préfixe de service	Actions
securitylake	lac de sécurité : CreateAwsLogSource lac de sécurité : CreateCustomLogSource lac de sécurité : CreateDataLakeExceptionSubscription lac de sécurité : CreateDataLakeOrganizationConfiguration lac de sécurité : CreateSubscriber lac de sécurité : CreateSubscriberNotification lac de sécurité : DeleteAwsLogSource lac de sécurité : DeleteCustomLogSource lac de sécurité : DeleteDataLakeExceptionSubscription lac de sécurité : DeleteDataLakeOrganizationConfiguration lac de sécurité : DeleteSubscriber lac de sécurité : DeleteSubscriberNotification lac de sécurité : DeregisterDataLakeDelegatedAdministrator lac de sécurité : GetDataLakeExceptionSubscription lac de sécurité : GetDataLakeOrganizationConfiguration lac de sécurité : GetDataLakeSources lac de sécurité : GetSubscriber lac de sécurité : ListDataLakes lac de sécurité : ListLogSources lac de sécurité : ListSubscribers lac de sécurité : RegisterDataLakeDelegatedAdministrator

Préfixe de service	Actions
	lac de sécurité : UpdateDataLakeExceptionSubscription lac de sécurité : UpdateSubscriber lac de sécurité : UpdateSubscriberNotification
serverlessrepo	dépôt sans serveur : CreateApplication dépôt sans serveur : CreateApplicationVersion dépôt sans serveur : CreateCloudFormationChangeSet dépôt sans serveur : CreateCloudFormationTemplate dépôt sans serveur : DeleteApplication dépôt sans serveur : GetApplication dépôt sans serveur : GetApplicationPolicy dépôt sans serveur : GetCloudFormationTemplate dépôt sans serveur : ListApplicationDependencies dépôt sans serveur : ListApplicationVersions dépôt sans serveur : ListApplications dépôt sans serveur : PutApplicationPolicy dépôt sans serveur : UnshareApplication dépôt sans serveur : UpdateApplication

Préfixe de service	Actions
servicecatalog	catalogue de services : AcceptPortfolioShare catalogue de services : AssociateBudgetWithResource catalogue de services : AssociatePrincipalWithPortfolio catalogue de services : AssociateProductWithPortfolio catalogue de services : AssociateServiceActionWithProvisioningArtifact catalogue de services : BatchAssociateServiceActionWithProvisioningArtifact catalogue de services : BatchDisassociateServiceActionFromProvisioningArtifact catalogue de services : CopyProduct catalogue de services : CreateAttributeGroup catalogue de services : CreateConstraint catalogue de services : CreatePortfolio catalogue de services : CreatePortfolioShare catalogue de services : CreateProduct catalogue de services : CreateProvisionedProductPlan catalogue de services : CreateProvisioningArtifact catalogue de services : CreateServiceAction catalogue de services : DeleteAttributeGroup catalogue de services : DeleteConstraint catalogue de services : DeletePortfolio

Préfixe de service	Actions
	catalogue de services : DeletePortfolioShare catalogue de services : DeleteProduct catalogue de services : DeleteProvisionedProductPlan catalogue de services : DeleteProvisioningArtifact catalogue de services : DeleteServiceAction catalogue de services : DescribeConstraint catalogue de services : DescribeCopyProductStatus catalogue de services : DescribePortfolio catalogue de services : DescribePortfolioShareStatus catalogue de services : DescribePortfolioShares catalogue de services : DescribeProduct catalogue de services : DescribeProductAsAdmin catalogue de services : DescribeProductView catalogue de services : DescribeProvisionedProduct catalogue de services : DescribeProvisionedProductPlan catalogue de services : DescribeProvisioningArtifact catalogue de services : DescribeProvisioningParameters catalogue de services : DescribeRecord catalogue de services : DescribeServiceAction catalogue de services : DescribeServiceActionExecutionParameters catalogue de services : DisableAWSOrganizationsAccess

Préfixe de service	Actions
	catalogue de services : DisassociateBudgetFromResource catalogue de services : DisassociatePrincipalFromPortfolio catalogue de services : DisassociateProductFromPortfolio catalogue de services : DisassociateServiceActionFromProvisioningArtifact catalogue de services : EnableAWSOrganizationsAccess catalogue de services : ExecuteProvisionedProductPlan catalogue de services : ExecuteProvisionedProductServiceAction catalogue de services : GetAWSOrganizationsAccessStatus catalogue de services : GetProvisionedProductOutputs catalogue de services : ImportAsProvisionedProduct catalogue de services : ListAcceptedPortfolioShares catalogue de services : ListAttributeGroups catalogue de services : ListBudgetsForResource catalogue de services : ListConstraintsForPortfolio catalogue de services : ListLaunchPaths catalogue de services : ListOrganizationPortfolioAccess catalogue de services : ListPortfolioAccess catalogue de services : ListPortfolios catalogue de services : ListPortfoliosForProduct catalogue de services : ListPrincipalsForPortfolio

Préfixe de service	Actions
	catalogue de services : ListProvisionedProductPlans catalogue de services : ListProvisioningArtifacts catalogue de services : ListProvisioningArtifactsForServiceAction catalogue de services : ListRecordHistory catalogue de services : ListServiceActions catalogue de services : ListServiceActionsForProvisioningArtifact catalogue de services : ListStackInstancesForProvisionedProduct catalogue de services : NotifyProvisionProductEngineWorkflow Result catalogue de services : NotifyTerminateProvisionedProductEng ineWorkflowResult catalogue de services : NotifyUpdateProvisionedProductEngine WorkflowResult catalogue de services : ProvisionProduct catalogue de services : RejectPortfolioShare catalogue de services : ScanProvisionedProducts catalogue de services : SearchProducts catalogue de services : SearchProductsAsAdmin catalogue de services : SearchProvisionedProducts catalogue de services : TerminateProvisionedProduct catalogue de services : UpdateConstraint catalogue de services : UpdatePortfolio

Préfixe de service	Actions
	catalogue de services : UpdatePortfolioShare catalogue de services : UpdateProduct catalogue de services : UpdateProvisionedProduct catalogue de services : UpdateProvisionedProductProperties catalogue de services : UpdateProvisioningArtifact catalogue de services : UpdateServiceAction

Préfixe de service	Actions
servicediscovery	découverte des services : CreateHttpNamespace
	découverte des services : CreatePrivateDnsNamespace
	découverte des services : CreatePublicDnsNamespace
	découverte des services : CreateService
	découverte des services : DeleteNamespace
	découverte des services : DeleteService
	découverte des services : DeleteServiceAttributes
	découverte des services : DeregisterInstance
	découverte des services : GetInstance
	découverte des services : GetInstancesHealthStatus
	découverte des services : GetNamespace
	découverte des services : GetOperation
	découverte des services : GetService
	découverte des services : ListInstances
	découverte des services : ListNamespaces
	découverte des services : ListOperations
	découverte des services : ListServices
	découverte des services : RegisterInstance
	découverte des services : UpdateHttpNamespace
	découverte des services : UpdateInstanceCustomHealthStatus
	découverte des services : UpdatePrivateDnsNamespace

Préfixe de service	Actions
	découverte des services : UpdatePublicDnsNamespace découverte des services : UpdateService découverte des services : UpdateServiceAttributes

Préfixe de service	Actions
servicequotas	devis de service : AssociateServiceQuotaTemplate devis de service : CreateSupportCase devis de service : DeleteServiceQuotaIncreaseRequestFromTemplate devis de service : DisassociateServiceQuotaTemplate devis de service : GetAWSDefaultServiceQuota devis de service : GetAssociationForServiceQuotaTemplate devis de service : GetAutoManagementConfiguration devis de service : GetQuotaUtilizationReport devis de service : GetRequestedServiceQuotaChange devis de service : GetServiceQuota devis de service : GetServiceQuotaIncreaseRequestFromTemplate devis de service : ListAWSDefaultServiceQuotas devis de service : ListRequestedServiceQuotaChangeHistory devis de service : ListRequestedServiceQuotaChangeHistoryByQuota devis de service : ListServiceQuotaIncreaseRequestsInTemplate devis de service : ListServiceQuotas devis de service : ListServices devis de service : PutServiceQuotaIncreaseRequestIntoTemplate devis de service : RequestServiceQuotaIncrease devis de service : StartAutoManagement

Préfixe de service	Actions
	devis de service : StartQuotaUtilizationReport devis de service : StopAutoManagement devis de service : UpdateAutoManagement

Préfixe de service	Actions
ses	Utilise : BatchGetMetricData
	Utilise : CloneReceiptRuleSet
	Utilise : CreateAddonInstance
	Utilise : CreateAddonSubscription
	Utilise : CreateAddressList
	Utilise : CreateAddressListImportJob
	Utilise : CreateArchive
	Utilise : CreateConfigurationSet
	Utilise : CreateConfigurationSetEventDestination
	Utilise : CreateConfigurationSetTrackingOptions
	Utilise : CreateContact
	Utilise : CreateContactList
	Utilise : CreateCustomVerificationEmailTemplate
	Utilise : CreateDedicatedIpPool
	Utilise : CreateDeliverabilityTestReport
	Utilise : CreateEmailIdentity
	Utilise : CreateEmailIdentityPolicy
	Utilise : CreateEmailTemplate
	Utilise : CreateImportJob
	Utilise : CreateIngressPoint
	Utilise : CreateMultiRegionEndpoint

Préfixe de service	Actions
	Utilise : CreateReceiptFilter
	Utilise : CreateReceiptRule
	Utilise : CreateReceiptRuleSet
	Utilise : CreateRelay
	Utilise : CreateRuleSet
	Utilise : CreateTemplate
	Utilise : CreateTenant
	Utilise : CreateTenantResourceAssociation
	Utilise : CreateTrafficPolicy
	Utilise : DeleteAddonInstance
	Utilise : DeleteAddonSubscription
	Utilise : DeleteAddressList
	Utilise : DeleteArchive
	Utilise : DeleteConfigurationSet
	Utilise : DeleteConfigurationSetEventDestination
	Utilise : DeleteConfigurationSetTrackingOptions
	Utilise : DeleteContact
	Utilise : DeleteContactList
	Utilise : DeleteCustomVerificationEmailTemplate
	Utilise : DeleteDedicatedIpPool
	Utilise : DeleteEmailIdentity

Préfixe de service	Actions
	Utilise : DeleteEmailIdentityPolicy
	Utilise : DeleteEmailTemplate
	Utilise : DeleteIdentity
	Utilise : DeleteIdentityPolicy
	Utilise : DeleteIngressPoint
	Utilise : DeleteMultiRegionEndpoint
	Utilise : DeleteReceiptFilter
	Utilise : DeleteReceiptRule
	Utilise : DeleteReceiptRuleSet
	Utilise : DeleteRelay
	Utilise : DeleteRuleSet
	Utilise : DeleteSuppressedDestination
	Utilise : DeleteTemplate
	Utilise : DeleteTenant
	Utilise : DeleteTenantResourceAssociation
	Utilise : DeleteTrafficPolicy
	Utilise : DeleteVerifiedEmailAddress
	Utilise : DeregisterMemberFromAddressList
	Utilise : DescribeActiveReceiptRuleSet
	Utilise : DescribeConfigurationSet
	Utilise : DescribeReceiptRule

Préfixe de service	Actions
	Utilise : DescribeReceiptRuleSet
	Utilise : GetAccount
	Utilise : GetAccountSendingEnabled
	Utilise : GetAddonInstance
	Utilise : GetAddonSubscription
	Utilise : GetAddressList
	Utilise : GetArchive
	Utilise : GetArchiveExport
	Utilise : GetArchiveMessage
	Utilise : GetArchiveMessageContent
	Utilise : GetArchiveSearch
	Utilise : GetArchiveSearchResults
	Utilise : GetBlacklistReports
	Utilise : GetConfigurationSet
	Utilise : GetConfigurationSetEventDestinations
	Utilise : GetContact
	Utilise : GetContactList
	Utilise : GetCustomVerificationEmailTemplate
	Utilise : GetDedicatedIp
	Utilise : GetDedicatedIpPool
	Utilise : GetDedicatedIps

Préfixe de service	Actions
	Utilise : GetDeliverabilityDashboardOptions
	Utilise : GetDeliverabilityTestReport
	Utilise : GetDomainDeliverabilityCampaign
	Utilise : GetDomainStatisticsReport
	Utilise : GetEmailAddressInsights
	Utilise : GetEmailIdentity
	Utilise : GetEmailIdentityPolicies
	Utilise : GetEmailTemplate
	Utilise : GetIdentityDkimAttributes
	Utilise : GetIdentityMailFromDomainAttributes
	Utilise : GetIdentityNotificationAttributes
	Utilise : GetIdentityPolicies
	Utilise : GetIdentityVerificationAttributes
	Utilise : GetImportJob
	Utilise : GetIngressPoint
	Utilise : GetMemberOfAddressList
	Utilise : GetMessageInsights
	Utilise : GetMultiRegionEndpoint
	Utilise : GetRelay
	Utilise : GetRuleSet
	Utilise : GetSendQuota

Préfixe de service	Actions
	Utilise : GetSendStatistics
	Utilise : GetSuppressedDestination
	Utilise : GetTemplate
	Utilise : GetTenant
	Utilise : GetTrafficPolicy
	Utilise : ListAddonInstances
	Utilise : ListAddonSubscriptions
	Utilise : ListAddressListImportJobs
	Utilise : ListAddressLists
	Utilise : ListArchiveExports
	Utilise : ListArchiveSearches
	Utilise : ListArchives
	Utilise : ListConfigurationSets
	Utilise : ListContactLists
	Utilise : ListContacts
	Utilise : ListCustomVerificationEmailTemplates
	Utilise : ListDedicatedIpPools
	Utilise : ListDeliverabilityTestReports
	Utilise : ListDomainDeliverabilityCampaigns
	Utilise : ListEmailIdentities
	Utilise : ListEmailTemplates

Préfixe de service	Actions
	Utilise : ListExportJobs
	Utilise : ListIdentities
	Utilise : ListIdentityPolicies
	Utilise : ListImportJobs
	Utilise : ListIngressPoints
	Utilise : ListMembersOfAddressList
	Utilise : ListMultiRegionEndpoints
	Utilise : ListReceiptFilters
	Utilise : ListReceiptRuleSets
	Utilise : ListRecommendations
	Utilise : ListRelays
	Utilise : ListReputationEntities
	Utilise : ListResourceTenants
	Utilise : ListRuleSets
	Utilise : ListSuppressedDestinations
	Utilise : ListTemplates
	Utilise : ListTenantResources
	Utilise : ListTenants
	Utilise : ListTrafficPolicies
	Utilise : ListVerifiedEmailAddresses
	Utilise : PutAccountDedicatedIpWarmupAttributes

Préfixe de service	Actions
	Utilise : PutAccountDetails
	Utilise : PutAccountSendingAttributes
	Utilise : PutAccountSuppressionAttributes
	Utilise : PutAccountVdmAttributes
	Utilise : PutConfigurationSetArchivingOptions
	Utilise : PutConfigurationSetDeliveryOptions
	Utilise : PutConfigurationSetReputationOptions
	Utilise : PutConfigurationSetSendingOptions
	Utilise : PutConfigurationSetSuppressionOptions
	Utilise : PutConfigurationSetTrackingOptions
	Utilise : PutConfigurationSetVdmOptions
	Utilise : PutDedicatedIpInPool
	Utilise : PutDedicatedIpPoolScalingAttributes
	Utilise : PutDedicatedIpWarmupAttributes
	Utilise : PutDeliverabilityDashboardOption
	Utilise : PutEmailIdentityConfigurationSetAttributes
	Utilise : PutEmailIdentityDkimAttributes
	Utilise : PutEmailIdentityDkimSigningAttributes
	Utilise : PutEmailIdentityFeedbackAttributes
	Utilise : PutEmailIdentityMailFromAttributes
	Utilise : PutIdentityPolicy

Préfixe de service	Actions
	Utilise : PutSuppressedDestination
	Utilise : PutTenantSuppressionAttributes
	Utilise : RegisterMemberToAddressList
	Utilise : ReorderReceiptRuleSet
	Utilise : SendBounce
	Utilise : SendCustomVerificationEmail
	Utilise : SetActiveReceiptRuleSet
	Utilise : SetIdentityDkimEnabled
	Utilise : SetIdentityFeedbackForwardingEnabled
	Utilise : SetIdentityHeadersInNotificationsEnabled
	Utilise : SetIdentityMailFromDomain
	Utilise : SetIdentityNotificationTopic
	Utilise : SetReceiptRulePosition
	Utilise : StartArchiveExport
	Utilise : StartArchiveSearch
	Utilise : StopArchiveExport
	Utilise : StopArchiveSearch
	Utilise : TestRenderEmailTemplate
	Utilise : TestRenderTemplate
	Utilise : UpdateAccountSendingEnabled
	Utilise : UpdateArchive

Préfixe de service	Actions
	Utilise : UpdateConfigurationSetEventDestination
	Utilise : UpdateConfigurationSetReputationMetricsEnabled
	Utilise : UpdateConfigurationSetSendingEnabled
	Utilise : UpdateConfigurationSetTrackingOptions
	Utilise : UpdateContact
	Utilise : UpdateContactList
	Utilise : UpdateCustomVerificationEmailTemplate
	Utilise : UpdateEmailIdentityPolicy
	Utilise : UpdateEmailTemplate
	Utilise : UpdateIngressPoint
	Utilise : UpdateReceiptRule
	Utilise : UpdateRelay
	Utilise : UpdateRuleSet
	Utilise : UpdateTemplate
	Utilise : UpdateTrafficPolicy
	Utilise : VerifyDomainDkim
	Utilise : VerifyDomainIdentity
	Utilise : VerifyEmailAddress
	Utilise : VerifyEmailIdentity

Préfixe de service	Actions
shield	bouclier : AssociateDRTLogBucket bouclier : AssociateHealthCheck bouclier : AssociateProactiveEngagementDetails bouclier : CreateProtection bouclier : CreateProtectionGroup bouclier : CreateSubscription bouclier : DeleteProtection bouclier : DeleteProtectionGroup bouclier : DeleteSubscription bouclier : DescribeAttack bouclier : DescribeAttackStatistics shield:DescribeDRTAccess bouclier : DescribeEmergencyContactSettings bouclier : DescribeProtection bouclier : DescribeProtectionGroup bouclier : DescribeSubscription bouclier : DisableApplicationLayerAutomaticResponse bouclier : DisableProactiveEngagement bouclier : DisassociateDRTLogBucket shield:DisassociateDRTRole bouclier : DisassociateHealthCheck

Préfixe de service	Actions
	bouclier : EnableApplicationLayerAutomaticResponse bouclier : EnableProactiveEngagement bouclier : GetSubscriptionState bouclier : ListAttacks bouclier : ListProtectionGroups bouclier : ListProtections bouclier : ListResourcesInProtectionGroup bouclier : UpdateApplicationLayerAutomaticResponse bouclier : UpdateEmergencyContactSettings bouclier : UpdateProtectionGroup bouclier : UpdateSubscription

Préfixe de service	Actions
signer	signataire : AddProfilePermission signataire : CancelSigningProfile signataire : DescribeSigningJob signataire : GetSigningPlatform signataire : GetSigningProfile signataire : ListProfilePermissions signataire : ListSigningJobs signataire : ListSigningPlatforms signataire : ListSigningProfiles signataire : PutSigningProfile signataire : RemoveProfilePermission signataire : RevokeSignature signataire : RevokeSigningProfile signataire : SignPayload signataire : StartSigningJob

Préfixe de service	Actions
simspaceweaver	Simspaceweaver : CreateSnapshot Simspaceweaver : DeleteApp Simspaceweaver : DeleteSimulation Simspaceweaver : DescribeApp Simspaceweaver : DescribeSimulation Simspaceweaver : ListApps Simspaceweaver : ListSimulations Simspaceweaver : StartApp Simspaceweaver : StartClock Simspaceweaver : StartSimulation Simspaceweaver : StopApp Simspaceweaver : StopClock Simspaceweaver : StopSimulation

Préfixe de service	Actions
sms	SMS : CreateApp SMS : CreateReplicationJob SMS : DeleteApp SMS : DeleteAppLaunchConfiguration SMS : DeleteAppReplicationConfiguration SMS : DeleteAppValidationConfiguration SMS : DeleteReplicationJob SMS : DeleteServerCatalog SMS : DisassociateConnector SMS : GenerateChangeSet SMS : GenerateTemplate SMS : GetApp SMS : GetAppLaunchConfiguration SMS : GetAppReplicationConfiguration SMS : GetAppValidationConfiguration SMS : GetAppValidationOutput SMS : GetConnectors SMS : GetReplicationJobs SMS : GetReplicationRuns SMS : GetServers SMS : ImportAppCatalog

Préfixe de service	Actions
	SMS : ImportServerCatalog
	SMS : LaunchApp
	SMS : ListApps
	SMS : NotifyAppValidationOutput
	SMS : PutAppLaunchConfiguration
	SMS : PutAppReplicationConfiguration
	SMS : PutAppValidationConfiguration
	SMS : StartAppReplication
	SMS : StartOnDemandAppReplication
	SMS : StartOnDemandReplicationRun
	SMS : StopAppReplication
	SMS : TerminateApp
	SMS : UpdateApp
	SMS : UpdateReplicationJob

Préfixe de service	Actions
sms-voice	SMS et voix : AssociateProtectConfiguration SMS et voix : CreateConfigurationSet SMS et voix : CreateConfigurationSetEventDestination SMS et voix : CreateEventDestination SMS et voix : CreateOptOutList SMS et voix : CreatePool SMS et voix : CreateProtectConfiguration SMS et voix : CreateRegistration SMS et voix : CreateRegistrationAssociation SMS et voix : CreateRegistrationAttachment SMS et voix : CreateRegistrationVersion SMS et voix : CreateVerifiedDestinationNumber SMS et voix : DeleteAccountDefaultProtectConfiguration SMS et voix : DeleteConfigurationSet SMS et voix : DeleteConfigurationSetEventDestination SMS et voix : DeleteDefaultMessageType SMS et voix : DeleteDefaultSenderId SMS et voix : DeleteEventDestination SMS et voix : DeleteKeyword SMS et voix : DeleteMediaMessageSpendLimitOverride SMS et voix : DeleteOptOutList

Préfixe de service	Actions
	SMS et voix : DeleteOptedOutNumber SMS et voix : DeletePool SMS et voix : DeleteProtectConfiguration SMS et voix : DeleteProtectConfigurationRuleSetNumberOverride SMS et voix : DeleteRegistration SMS et voix : DeleteRegistrationAttachment SMS et voix : DeleteResourcePolicy SMS et voix : DeleteTextMessageSpendLimitOverride SMS et voix : DeleteVerifiedDestinationNumber SMS et voix : DeleteVoiceMessageSpendLimitOverride SMS et voix : DescribeAccountAttributes SMS et voix : DescribeAccountLimits SMS et voix : DescribeConfigurationSets SMS et voix : DescribeKeywords SMS et voix : DescribeOptOutLists SMS et voix : DescribeOptedOutNumbers SMS et voix : DescribePhoneNumbers SMS et voix : DescribePools SMS et voix : DescribeProtectConfigurations SMS et voix : DescribeRegistrationAttachments SMS et voix : DescribeRegistrationFieldDefinitions

Préfixe de service	Actions
	SMS et voix : DescribeRegistrationFieldValues
	SMS et voix : DescribeRegistrationSectionDefinitions
	SMS et voix : DescribeRegistrationTypeDefinitions
	SMS et voix : DescribeRegistrationVersions
	SMS et voix : DescribeRegistrations
	SMS et voix : DescribeSenderIds
	SMS et voix : DescribeSpendLimits
	SMS et voix : DescribeVerifiedDestinationNumbers
	SMS et voix : DisassociateOriginationIdentity
	SMS et voix : DisassociateProtectConfiguration
	SMS et voix : DiscardRegistrationVersion
	SMS et voix : GetConfigurationSetEventDestinations
	SMS et voix : GetProtectConfigurationCountryRuleSet
	SMS et voix : GetResourcePolicy
	SMS et voix : ListConfigurationSets
	SMS et voix : ListPoolOriginationIdentities
	SMS et voix : ListProtectConfigurationRuleSetNumberOverrides
	SMS et voix : ListRegistrationAssociations
	SMS et voix : PutKeyword
	SMS et voix : PutOptedOutNumber
	SMS et voix : PutProtectConfigurationRuleSetNumberOverride

Préfixe de service	Actions
	SMS et voix : PutResourcePolicy SMS et voix : ReleasePhoneNumber SMS et voix : ReleaseSenderId SMS et voix : RequestPhoneNumber SMS et voix : RequestSenderId SMS et voix : SendDestinationNumberVerificationCode SMS et voix : SetAccountDefaultProtectConfiguration SMS et voix : SetDefaultMessageFeedbackEnabled SMS et voix : SetDefaultMessageType SMS et voix : SetDefaultSenderId SMS et voix : SetMediaMessageSpendLimitOverride SMS et voix : SetTextMessageSpendLimitOverride SMS et voix : SetVoiceMessageSpendLimitOverride SMS et voix : SubmitRegistrationVersion SMS et voix : UpdateConfigurationSetEventDestination SMS et voix : UpdateEventDestination SMS et voix : UpdatePhoneNumber SMS et voix : UpdatePool SMS et voix : UpdateProtectConfiguration SMS et voix : UpdateProtectConfigurationCountryRuleSet SMS et voix : UpdateSenderId

Préfixe de service	Actions
snowball	boule de neige : CancelCluster
	boule de neige : CancelJob
	boule de neige : CreateAddress
	boule de neige : CreateCluster
	boule de neige : CreateJob
	boule de neige : CreateLongTermPricing
	boule de neige : CreateReturnShippingLabel
	boule de neige : DescribeAddress
	boule de neige : DescribeAddresses
	boule de neige : DescribeCluster
	boule de neige : DescribeJob
	boule de neige : DescribeReturnShippingLabel
	boule de neige : GetJobManifest
	boule de neige : GetJobUnlockCode
	boule de neige : GetSnowballUsage
	boule de neige : GetSoftwareUpdates
	boule de neige : ListClusterJobs
	boule de neige : ListClusters
	boule de neige : ListCompatibleImages
	boule de neige : ListJobs
	boule de neige : ListLongTermPricing

Préfixe de service	Actions
	boule de neige : ListPickupLocations boule de neige : ListServiceVersions boule de neige : UpdateCluster boule de neige : UpdateJob boule de neige : UpdateJobShipmentState boule de neige : UpdateLongTermPricing
sqs	sqs : AddPermission sqs : CancelMessageMoveTask sqs : CreateQueue sqs : DeleteQueue sqs : PurgeQueue sqs : RemovePermission sqs : SetQueueAttributes

Préfixe de service	Actions
ssm	SMS : AssociateOpsItemRelatedItem
	SMS : CancelCommand
	SMS : CancelMaintenanceWindowExecution
	SMS : CreateActivation
	SMS : CreateAssociation
	SMS : CreateAssociationBatch
	SMS : CreateDocument
	SMS : CreateMaintenanceWindow
	SMS : CreateOpsItem
	SMS : CreateOpsMetadata
	SMS : CreatePatchBaseline
	SMS : CreateResourceDataSync
	SMS : DeleteActivation
	SMS : DeleteAssociation
	SMS : DeleteCloudConnector
	SMS : DeleteDocument
	SMS : DeleteInventory
	SMS : DeleteMaintenanceWindow
	SMS : DeleteOpsItem
	SMS : DeleteOpsMetadata
	SMS : DeleteParameter

Préfixe de service	Actions
	SMS : DeleteParameters
	SMS : DeletePatchBaseline
	SMS : DeleteResourceDataSync
	SMS : DeleteResourcePolicy
	SMS : DeregisterManagedInstance
	SMS : DeregisterPatchBaselineForPatchGroup
	SMS : DeregisterTargetFromMaintenanceWindow
	SMS : DeregisterTaskFromMaintenanceWindow
	SMS : DescribeActivations
	SMS : DescribeAssociation
	SMS : DescribeAssociationExecutionTargets
	SMS : DescribeAssociationExecutions
	SMS : DescribeAutomationExecutions
	SMS : DescribeAutomationStepExecutions
	SMS : DescribeAvailablePatches
	SMS : DescribeDocument
	SMS : DescribeDocumentParameters
	SMS : DescribeDocumentPermission
	SMS : DescribeEffectiveInstanceAssociations
	SMS : DescribeEffectivePatchesForPatchBaseline
	SMS : DescribeInstanceAssociationsStatus

Préfixe de service	Actions
	SMS : DescribeInstanceInformation
	SMS : DescribeInstancePatchStates
	SMS : DescribeInstancePatchStatesForPatchGroup
	SMS : DescribeInstancePatches
	SMS : DescribeInstanceProperties
	SMS : DescribeInventoryDeletions
	SMS : DescribeMaintenanceWindowExecutionTaskInvocations
	SMS : DescribeMaintenanceWindowExecutionTasks
	SMS : DescribeMaintenanceWindowExecutions
	SMS : DescribeMaintenanceWindowSchedule
	SMS : DescribeMaintenanceWindowTargets
	SMS : DescribeMaintenanceWindowTasks
	SMS : DescribeMaintenanceWindows
	SMS : DescribeMaintenanceWindowsForTarget
	SMS : DescribeOpsItems
	SMS : DescribeParameters
	SMS : DescribePatchBaselines
	SMS : DescribePatchGroupState
	SMS : DescribePatchGroups
	SMS : DescribePatchProperties
	SMS : DescribeSessions

Préfixe de service	Actions
	SMS : DisassociateOpsItemRelatedItem
	SMS : GetAccessToken
	SMS : GetAutomationExecution
	SMS : GetCalendarState
	SMS : GetCloudConnector
	SMS : GetCommandInvocation
	SMS : GetConnectionStatus
	SMS : GetDefaultPatchBaseline
	SMS : GetDeployablePatchSnapshotForInstance
	SMS : GetDocument
	SMS : GetExecutionPreview
	SMS : GetInventory
	SMS : GetInventorySchema
	SMS : GetMaintenanceWindow
	SMS : GetMaintenanceWindowExecution
	SMS : GetMaintenanceWindowExecutionTask
	SMS : GetMaintenanceWindowExecutionTaskInvocation
	SMS : GetMaintenanceWindowTask
	SMS : GetOpsItem
	SMS : GetOpsMetadata
	SMS : GetOpsSummary

Préfixe de service	Actions
	SMS : GetParameter
	SMS : GetParameterHistory
	SMS : GetParameters
	SMS : GetParametersByPath
	SMS : GetPatchBaseline
	SMS : GetPatchBaselineForPatchGroup
	SMS : GetResourcePolicies
	SMS : GetServiceSetting
	SMS : LabelParameterVersion
	SMS : ListAssociationVersions
	SMS : ListAssociations
	SMS : ListCloudConnectors
	SMS : ListCommandInvocations
	SMS : ListCommands
	SMS : ListComplianceItems
	SMS : ListComplianceSummaries
	SMS : ListDocumentMetadataHistory
	SMS : ListDocumentVersions
	SMS : ListDocuments
	SMS : ListInstanceAssociations
	SMS : ListInventoryEntries

Préfixe de service	Actions
	SMS : ListNodes
	SMS : ListNodesSummary
	SMS : ListOpsItemEvents
	SMS : ListOpsItemRelatedItems
	SMS : ListOpsMetadata
	SMS : ListResourceComplianceSummaries
	SMS : ListResourceDataSync
	SMS : ModifyDocumentPermission
	SMS : PutComplianceItems
	SMS : PutInventory
	SMS : PutParameter
	SMS : PutResourcePolicy
	SMS : RegisterDefaultPatchBaseline
	SMS : RegisterManagedInstance
	SMS : RegisterPatchBaselineForPatchGroup
	SMS : RegisterTargetWithMaintenanceWindow
	SMS : RegisterTaskWithMaintenanceWindow
	SMS : ResetServiceSetting
	SMS : ResumeSession
	SMS : SendAutomationSignal
	SMS : SendCommand

Préfixe de service	Actions
	SMS : StartAssociationsOnce
	SMS : StartAutomationExecution
	SMS : StartChangeRequestExecution
	SMS : StartSession
	SMS : StopAutomationExecution
	SMS : TerminateSession
	SMS : UnlabelParameterVersion
	SMS : UpdateAssociation
	SMS : UpdateAssociationStatus
	SMS : UpdateCloudConnector
	SMS : UpdateDocument
	SMS : UpdateDocumentDefaultVersion
	SMS : UpdateDocumentMetadata
	SMS : UpdateInstanceInformation
	SMS : UpdateMaintenanceWindow
	SMS : UpdateMaintenanceWindowTarget
	SMS : UpdateMaintenanceWindowTask
	SMS : UpdateManagedInstanceRole
	SMS : UpdateOpsItem
	SMS : UpdateOpsMetadata
	SMS : UpdatePatchBaseline

Préfixe de service	Actions
	SMS : UpdateResourceDataSync SMS : UpdateServiceSetting SMS : ValidateCloudConnector

Préfixe de service	Actions
ssm-incidents	Incidents SMS : BatchGetIncidentFindings
	Incidents SMS : CreateReplicationSet
	Incidents SMS : CreateResponsePlan
	Incidents SMS : CreateTimelineEvent
	Incidents SMS : DeleteIncidentRecord
	Incidents SMS : DeleteReplicationSet
	Incidents SMS : DeleteResourcePolicy
	Incidents SMS : DeleteResponsePlan
	Incidents SMS : DeleteTimelineEvent
	Incidents SMS : GetIncidentRecord
	Incidents SMS : GetReplicationSet
	Incidents SMS : GetResourcePolicies
	Incidents SMS : GetResponsePlan
	Incidents SMS : GetTimelineEvent
	Incidents SMS : ListIncidentFindings
	Incidents SMS : ListIncidentRecords
	Incidents SMS : ListRelatedItems
	Incidents SMS : ListReplicationSets
	Incidents SMS : ListResponsePlans
	Incidents SMS : ListTimelineEvents
	Incidents SMS : PutResourcePolicy

Préfixe de service	Actions
	Incidents SMS : StartIncident Incidents SMS : UpdateDeletionProtection Incidents SMS : UpdateIncidentRecord Incidents SMS : UpdateRelatedItems Incidents SMS : UpdateReplicationSet Incidents SMS : UpdateResponsePlan Incidents SMS : UpdateTimelineEvent

Préfixe de service	Actions
ssm-sap	SSM-SAP : BackupDatabase
	SSM-SAP : DeleteResourcePermission
	SSM-SAP : DeregisterApplication
	SSM-SAP : GetApplication
	SSM-SAP : GetComponent
	SSM-SAP : GetConfigurationCheckOperation
	SSM-SAP : GetDatabase
	SSM-SAP : GetOperation
	SSM-SAP : GetResourcePermission
	SSM-SAP : ListApplications
	SSM-SAP : ListComponents
	SSM-SAP : ListConfigurationCheckDefinitions
	SSM-SAP : ListConfigurationCheckOperations
	SSM-SAP : ListDatabases
	SSM-SAP : ListOperationEvents
	SSM-SAP : ListOperations
	SSM-SAP : ListSubCheckResults
	SSM-SAP : ListSubCheckRuleResults
	SSM-SAP : PutResourcePermission
	SSM-SAP : RegisterApplication
	SSM-SAP : RestoreDatabase

Préfixe de service	Actions
	SSM-SAP : StartApplication
	SSM-SAP : StartApplicationRefresh
	SSM-SAP : StartConfigurationChecks
	SSM-SAP : StopApplication
	SSM-SAP : UpdateApplicationSettings
	SSM-SAP : UpdateHANABackupSettings

Préfixe de service	Actions
states	états : CreateActivity états : CreateStateMachine états : CreateStateMachineAlias états : DeleteActivity états : DeleteStateMachine états : DeleteStateMachineAlias états : DeleteStateMachineVersion états : DescribeActivity états : DescribeExecution états : DescribeMapRun états : DescribeStateMachine états : DescribeStateMachineAlias états : DescribeStateMachineForExecution états : GetExecutionHistory états : ListActivities états : ListExecutions états : ListMapRuns états : ListStateMachineAliases états : ListStateMachineVersions états : ListStateMachines états : SendTaskFailure

Préfixe de service	Actions
	états : SendTaskHeartbeat
	états : SendTaskSuccess
	états : StartExecution
	états : StopExecution
	états : UpdateMapRun
	états : UpdateStateMachine
	états : UpdateStateMachineAlias
	états : ValidateStateMachineDefinition
sts	sets : AssumeRole
	sets : AssumeRoleWithSAML
	sets : AssumeRoleWithWebIdentity
	sets : DecodeAuthorizationMessage
	sets : GetAccessKeyInfo
	sets : GetCallerIdentity
	sets : GetFederationToken
	sets : GetSessionToken
	sets : GetWebIdentityToken

Préfixe de service	Actions
swf	swf : DeleteActivityType swf : DeleteWorkflowType swf : DeprecateActivityType swf : DeprecateDomain swf : DeprecateWorkflowType swf : DescribeActivityType swf : DescribeDomain swf : DescribeWorkflowType swf : ListActivityTypes swf : ListDomains swf : ListWorkflowTypes swf : RegisterActivityType swf : RegisterDomain swf : RegisterWorkflowType swf : UndeprecateActivityType swf : UndeprecateDomain swf : UndeprecateWorkflowType

Préfixe de service	Actions
synthetics	synthétiques : AssociateResource
	synthétiques : CreateCanary
	synthétiques : CreateGroup
	synthétiques : DeleteCanary
	synthétiques : DeleteGroup
	synthétiques : DescribeCanaries
	synthétiques : DescribeCanariesLastRun
	synthétiques : DescribeRuntimeVersions
	synthétiques : DisassociateResource
	synthétiques : GetCanary
	synthétiques : GetCanaryRuns
	synthétiques : GetGroup
	synthétiques : ListAssociatedGroups
	synthétiques : ListGroupResources
	synthétiques : ListGroups
	synthétiques : StartCanary
	synthétiques : StartCanaryDryRun
	synthétiques : StopCanary
	synthétiques : UpdateCanary

Préfixe de service	Actions
balise	étiquette : DescribeReportCreation étiquette : GetComplianceSummary étiquette : GetResources étiquette : StartReportCreation

Préfixe de service	Actions
textract	extrait : AnalyzeDocument
	extrait : AnalyzeExpense
	textract:AnalyzeID
	extrait : CreateAdapter
	extrait : CreateAdapterVersion
	extrait : DeleteAdapter
	extrait : DeleteAdapterVersion
	extrait : DetectDocumentText
	extrait : GetAdapter
	extrait : GetAdapterVersion
	extrait : GetDocumentAnalysis
	extrait : GetDocumentTextDetection
	extrait : GetExpenseAnalysis
	extrait : GetLendingAnalysis
	extrait : GetLendingAnalysisSummary
	extrait : ListAdapterVersions
	extrait : ListAdapters
	extrait : StartDocumentAnalysis
	extrait : StartDocumentTextDetection
	extrait : StartExpenseAnalysis
	extrait : StartLendingAnalysis

Préfixe de service	Actions
	extrait : UpdateAdapter

Préfixe de service	Actions
timestream	diffusion chronologique : CancelQuery diffusion chronologique : CreateDatabase diffusion chronologique : CreateScheduledQuery diffusion chronologique : CreateTable diffusion chronologique : DeleteDatabase diffusion chronologique : DeleteScheduledQuery diffusion temporelle : DeleteTable diffusion temporelle : DescribeAccountSettings diffusion temporelle : DescribeDatabase diffusion temporelle : DescribeScheduledQuery diffusion temporelle : DescribeTable diffusion temporelle : ExecuteScheduledQuery diffusion temporelle : ListBatchLoadTasks diffusion temporelle : ListDatabases diffusion temporelle : ListScheduledQueries diffusion temporelle : ListTables diffusion temporelle : PrepareQuery diffusion temporelle : UpdateAccountSettings diffusion temporelle : UpdateDatabase diffusion temporelle : UpdateScheduledQuery diffusion temporelle : UpdateTable

Préfixe de service	Actions
tnb	tnb : CancelSolNetworkOperation
	tnb : CreateSolFunctionPackage
	tnb : CreateSolNetworkInstance
	tnb : CreateSolNetworkPackage
	tnb : DeleteSolFunctionPackage
	tnb : DeleteSolNetworkInstance
	tnb : DeleteSolNetworkPackage
	tnb : GetSolFunctionInstance
	tnb : GetSolFunctionPackage
	tnb : GetSolFunctionPackageContent
	tnb : GetSolFunctionPackageDescriptor
	tnb : GetSolNetworkInstance
	tnb : GetSolNetworkOperation
	tnb : GetSolNetworkPackage
	tnb : GetSolNetworkPackageContent
	tnb : GetSolNetworkPackageDescriptor
	tnb : InstantiateSolNetworkInstance
	tnb : ListSolFunctionInstances
	tnb : ListSolFunctionPackages
	tnb : ListSolNetworkInstances
	tnb : ListSolNetworkOperations

Préfixe de service	Actions
	tnb : ListSolNetworkPackages tnb : PutSolFunctionPackageContent tnb : PutSolNetworkPackageContent tnb : TerminateSolNetworkInstance tnb : UpdateSolFunctionPackage tnb : UpdateSolNetworkInstance tnb : UpdateSolNetworkPackage tnb : ValidateSolFunctionPackageContent tnb : ValidateSolNetworkPackageContent

Préfixe de service	Actions
transcribe	transcrire : CreateCallAnalyticsCategory transcrire : CreateLanguageModel transcrire : CreateMedicalVocabulary transcrire : CreateVocabulary transcrire : CreateVocabularyFilter transcrire : DeleteCallAnalyticsCategory transcrire : DeleteCallAnalyticsJob transcrire : DeleteLanguageModel transcrire : DeleteMedicalScribeJob transcrire : DeleteMedicalTranscriptionJob transcrire : DeleteMedicalVocabulary transcrire : DeleteTranscriptionJob transcrire : DeleteVocabulary transcrire : DeleteVocabularyFilter transcrire : DescribeLanguageModel transcrire : GetCallAnalyticsCategory transcrire : GetCallAnalyticsJob transcrire : GetMedicalScribeJob transcrire : GetMedicalTranscriptionJob transcrire : GetMedicalVocabulary transcrire : GetTranscriptionJob

Préfixe de service	Actions
	transcrire : GetVocabulary transcrire : GetVocabularyFilter transcrire : ListCallAnalyticsCategories transcrire : ListCallAnalyticsJobs transcrire : ListLanguageModels transcrire : ListMedicalScribeJobs transcrire : ListMedicalTranscriptionJobs transcrire : ListMedicalVocabularies transcrire : ListTranscriptionJobs transcrire : ListVocabularies transcrire : ListVocabularyFilters transcrire : StartCallAnalyticsJob transcrire : StartCallAnalyticsStreamTranscription transcrire : StartCallAnalyticsStreamTranscriptionWebSocket transcrire : StartMedicalScribeJob transcrire : StartMedicalStreamTranscription transcrire : StartMedicalStreamTranscriptionWebSocket transcrire : StartMedicalTranscriptionJob transcrire : StartStreamTranscription transcrire : StartStreamTranscriptionWebSocket transcrire : StartTranscriptionJob

Préfixe de service	Actions
	transcrire : UpdateCallAnalyticsCategory transcrire : UpdateMedicalVocabulary transcrire : UpdateVocabulary transcrire : UpdateVocabularyFilter

Préfixe de service	Actions
transfert	transfert : CreateAccess
	transfert : CreateAgreement
	transfert : CreateConnector
	transfert : CreateProfile
	transfert : CreateServer
	transfert : CreateUser
	transfert : CreateWebApp
	transfert : CreateWorkflow
	transfert : DeleteAccess
	transfert : DeleteAgreement
	transfert : DeleteCertificate
	transfert : DeleteConnector
	transfert : DeleteHostKey
	transfert : DeleteProfile
	transfert : DeleteServer
	transfert : DeleteSshPublicKey
	transfert : DeleteUser
	transfert : DeleteWebApp
	transfert : DeleteWebAppCustomization
	transfert : DeleteWorkflow
	transfert : DescribeAccess

Préfixe de service	Actions
	transfert : DescribeAgreement
	transfert : DescribeCertificate
	transfert : DescribeConnector
	transfert : DescribeExecution
	transfert : DescribeHostKey
	transfert : DescribeProfile
	transfert : DescribeSecurityPolicy
	transfert : DescribeServer
	transfert : DescribeUser
	transfert : DescribeWebApp
	transfert : DescribeWebAppCustomization
	transfert : DescribeWorkflow
	transfert : ImportCertificate
	transfert : ImportHostKey
	transfert : ImportSshPublicKey
	transfert : ListAccesses
	transfert : ListCertificates
	transfert : ListConnectors
	transfert : ListExecutions
	transfert : ListFileTransferResults
	transfert : ListHostKeys

Préfixe de service	Actions
	transfert : ListProfiles
	transfert : ListSecurityPolicies
	transfert : ListServers
	transfert : ListUsers
	transfert : ListWebApps
	transfert : ListWorkflows
	transfert : SendWorkflowStepState
	transfert : StartDirectoryListing
	transfert : StartFileTransfer
	transfert : StartRemoteDelete
	transfert : StartRemoteMove
	transfert : StartServer
	transfert : StopServer
	transfert : TestConnection
	transfert : TestIdentityProvider
	transfert : UpdateAccess
	transfert : UpdateAgreement
	transfert : UpdateCertificate
	transfert : UpdateConnector
	transfert : UpdateHostKey
	transfert : UpdateProfile

Préfixe de service	Actions
	transfert : UpdateServer transfert : UpdateUser transfert : UpdateWebApp transfert : UpdateWebAppCustomization
translate	traduire : CreateParallelData traduire : DeleteParallelData traduire : DeleteTerminology traduire : DescribeTextTranslationJob traduire : GetParallelData traduire : GetTerminology traduire : ImportTerminology traduire : ListLanguages traduire : ListParallelData traduire : ListTerminologies traduire : ListTextTranslationJobs traduire : StartTextTranslationJob traduire : StopTextTranslationJob traduire : TranslateDocument traduire : TranslateText traduire : UpdateParallelData

Préfixe de service	Actions
voiceid	identifiant vocal : AssociateFraudster
	identifiant vocal : CreateDomain
	identifiant vocal : CreateWatchlist
	identifiant vocal : DeleteDomain
	identifiant vocal : DeleteFraudster
	identifiant vocal : DeleteSpeaker
	identifiant vocal : DeleteWatchlist
	identifiant vocal : DescribeDomain
	identifiant vocal : DescribeFraudster
	identifiant vocal : DescribeFraudsterRegistrationJob
	identifiant vocal : DescribeSpeaker
	identifiant vocal : DescribeSpeakerEnrollmentJob
	identifiant vocal : DescribeWatchlist
	identifiant vocal : DisassociateFraudster
	identifiant vocal : EvaluateSession
	identifiant vocal : ListDomains
	identifiant vocal : ListFraudsterRegistrationJobs
	identifiant vocal : ListFraudsters
	identifiant vocal : ListSpeakerEnrollmentJobs
	identifiant vocal : ListSpeakers
	identifiant vocal : ListWatchlists

Préfixe de service	Actions
	identifiant vocal : OptOutSpeaker
	identifiant vocal : StartFraudsterRegistrationJob
	identifiant vocal : StartSpeakerEnrollmentJob
	identifiant vocal : UpdateDomain
	identifiant vocal : UpdateWatchlist

Préfixe de service	Actions
vpc-lattice	réseau VPC : CreateAccessLogSubscription réseau VPC : CreateListener réseau VPC : CreateResourceConfiguration réseau VPC : CreateResourceGateway réseau VPC : CreateRule réseau VPC : CreateService réseau VPC : CreateServiceNetwork réseau VPC : CreateServiceNetworkResourceAssociation réseau VPC : CreateServiceNetworkServiceAssociation réseau VPC : CreateServiceNetworkVpcAssociation réseau VPC : CreateTargetGroup réseau VPC : DeleteAccessLogSubscription réseau VPC : DeleteAuthPolicy réseau VPC : DeleteDomainVerification réseau VPC : DeleteListener réseau VPC : DeleteResourceConfiguration réseau VPC : DeleteResourceEndpointAssociation réseau VPC : DeleteResourceGateway réseau VPC : DeleteResourcePolicy réseau VPC : DeleteRule réseau VPC : DeleteService

Préfixe de service	Actions
	réseau VPC : DeleteServiceNetwork réseau VPC : DeleteServiceNetworkResourceAssociation réseau VPC : DeleteServiceNetworkServiceAssociation réseau VPC : DeleteServiceNetworkVpcAssociation réseau VPC : DeleteTargetGroup réseau VPC : DeregisterTargets réseau VPC : GetAccessLogSubscription réseau VPC : GetAuthPolicy réseau VPC : GetDomainVerification réseau VPC : GetListener réseau VPC : GetResourceConfiguration réseau VPC : GetResourceGateway réseau VPC : GetResourcePolicy réseau VPC : GetRule réseau VPC : GetService réseau VPC : GetServiceNetwork réseau VPC : GetServiceNetworkResourceAssociation réseau VPC : GetServiceNetworkServiceAssociation réseau VPC : GetServiceNetworkVpcAssociation réseau VPC : GetTargetGroup réseau VPC : ListAccessLogSubscriptions

Préfixe de service	Actions
	réseau VPC : ListDomainVerifications réseau VPC : ListListeners réseau VPC : ListResourceConfigurations réseau VPC : ListResourceEndpointAssociations réseau VPC : ListResourceGateways réseau VPC : ListRules réseau VPC : ListServiceNetworkResourceAssociations réseau VPC : ListServiceNetworkServiceAssociations réseau VPC : ListServiceNetworkVpcAssociations réseau VPC : ListServiceNetworkVpcEndpointAssociations réseau VPC : ListServiceNetworks réseau VPC : ListServices réseau VPC : ListTargetGroups réseau VPC : ListTargets réseau VPC : PutAuthPolicy réseau VPC : PutResourcePolicy réseau VPC : RegisterTargets réseau VPC : StartDomainVerification réseau VPC : UpdateAccessLogSubscription réseau VPC : UpdateListener réseau VPC : UpdateResourceConfiguration

Préfixe de service	Actions
	réseau VPC : UpdateResourceGateway réseau VPC : UpdateRule réseau VPC : UpdateService réseau VPC : UpdateServiceNetwork réseau VPC : UpdateServiceNetworkVpcAssociation réseau VPC : UpdateTargetGroup

Préfixe de service	Actions
wafv2	wafv2 : ACL AssociateWeb
	wafv2 : CheckCapacity
	wafv2:CreateAPIKey
	wafv2:CreateIPSet
	wafv2 : CreateRegexPatternSet
	wafv2 : CreateRuleGroup
	wafv2 : ACL CreateWeb
	wafv2:DeleteAPIKey
	wafv2 : DeleteFirewallManagerRuleGroups
	wafv2:DeleteIPSet
	wafv2 : DeleteLoggingConfiguration
	wafv2 : DeletePermissionPolicy
	wafv2 : DeleteRegexPatternSet
	wafv2 : DeleteRuleGroup
	wafv2 : ACL DeleteWeb
	wafv2 : DescribeAllManagedProducts
	wafv2 : DescribeManagedProductsByVendor
	wafv2 : DescribeManagedRuleGroup
	wafv2 : ACL DisassociateWeb
	wafv2 : GenerateMobileSdkReleaseUrl
	wafv2 : GetDecryptedAPIKey

Préfixe de service	Actions
	wafv2:GetIPSet
	wafv2 : GetLoggingConfiguration
	wafv2 : GetManagedRuleSet
	wafv2 : GetMobileSdkRelease
	wafv2 : GetRateBasedStatementManagedKeys
	wafv2 : GetRegexPatternSet
	wafv2 : GetRevenueStatistics
	wafv2 : GetRevenueStatisticsSummary
	wafv2 : GetRevenueStatisticsTimeSeries
	wafv2 : GetRuleGroup
	wafv2 : GetSampledRequests
	wafv2 : GetWebACLForResource
	wafv2:ListAPIKeys
	wafv2 : ListAvailableManagedRuleGroupVersions
	wafv2 : ListAvailableManagedRuleGroups
	wafv2:ListIPSets
	wafv2 : ListLoggingConfigurations
	wafv2 : ListManagedRuleSets
	wafv2 : ListMobileSdkReleases
	wafv2 : ListRegexPatternSets
	wafv2 : ListResourcesForWebACL

Préfixe de service	Actions
	wafv2 : ListRuleGroups wafv2 : ListSettlementRecords wafv2 : ListWebACLs wafv2 : PutLoggingConfiguration wafv2 : PutManagedRuleSetVersions wafv2:UpdateIPSet wafv2 : UpdateManagedRuleSetVersionExpiryDate wafv2 : UpdateRegexPatternSet wafv2 : UpdateRuleGroup wafv2 : ACL UpdateWeb

Préfixe de service	Actions
wellarchitected	bien architecturé : AssociateLenses bien architecturé : AssociateProfiles bien architecturé : CreateLensShare bien architecturé : CreateLensVersion bien architecturé : CreateMilestone bien architecturé : CreateProfile bien architecturé : CreateProfileShare bien architecturé : CreateReviewTemplate bien architecturé : CreateWorkload bien architecturé : CreateWorkloadShare bien architecturé : DeleteLens bien architecturé : DeleteLensShare bien architecturé : DeleteProfile bien architecturé : DeleteProfileShare bien architecturé : DeleteReviewTemplate bien architecturé : DeleteTemplateShare bien architecturé : DeleteWorkload bien architecturé : DeleteWorkloadShare bien architecturé : DisassociateLenses bien architecturé : DisassociateProfiles bien architecturé : ExportLens

Préfixe de service	Actions
	bien architecturé : GetAnswer bien architecturé : GetConsolidatedReport bien architecturé : GetGlobalSettings bien architecturé : GetLens bien architecturé : GetLensReview bien architecturé : GetLensReviewReport bien architecturé : GetLensVersionDifference bien architecturé : GetMilestone bien architecturé : GetProfile bien architecturé : GetProfileTemplate bien architecturé : GetReviewTemplate bien architecturé : GetReviewTemplateAnswer bien architecturé : GetReviewTemplateLensReview bien architecturé : GetWorkload bien architecturé : ImportLens bien architecturé : ListAnswers bien architecturé : ListCheckDetails bien architecturé : ListCheckSummaries bien architecturé : ListLensReviewImprovements bien architecturé : ListLensReviews bien architecturé : ListLensShares

Préfixe de service	Actions
	bien architecturé : ListLenses bien architecturé : ListMilestones bien architecturé : ListNotifications bien architecturé : ListProfileNotifications bien architecturé : ListProfileShares bien architecturé : ListProfiles bien architecturé : ListReviewTemplateAnswers bien architecturé : ListReviewTemplates bien architecturé : ListShareInvitations bien architecturé : ListTemplateShares bien architecturé : ListWorkloadShares bien architecturé : ListWorkloads bien architecturé : UpdateAnswer bien architecturé : UpdateGlobalSettings bien conçu : UpdateIntegration bien conçu : UpdateLensReview bien conçu : UpdateProfile bien conçu : UpdateReviewTemplate bien conçu : UpdateReviewTemplateLensReview bien conçu : UpdateShareInvitation bien conçu : UpdateWorkload

Préfixe de service	Actions
	bien conçu : UpdateWorkloadShare bien conçu : UpgradeLensReview bien conçu : UpgradeProfileVersion bien conçu : UpgradeReviewTemplateLensReview

Préfixe de service	Actions
wisdom	sagesse : CreateAssistant
	sagesse : CreateAssistantAssociation
	sagesse : CreateContent
	sagesse : CreateKnowledgeBase
	sagesse : CreateQuickResponse
	sagesse : CreateSession
	sagesse : DeleteAssistant
	sagesse : DeleteAssistantAssociation
	sagesse : DeleteContent
	sagesse : DeleteImportJob
	sagesse : DeleteKnowledgeBase
	sagesse : DeleteQuickResponse
	sagesse : GetAssistant
	sagesse : GetAssistantAssociation
	sagesse : GetContent
	sagesse : GetContentAssociation
	sagesse : GetContentSummary
	sagesse : GetImportJob
	sagesse : GetKnowledgeBase
	sagesse : GetRecommendations
	sagesse : GetSession

Préfixe de service	Actions
	sagesse : ListAssistantAssociations
	sagesse : ListAssistants
	sagesse : ListContentAssociations
	sagesse : ListContents
	sagesse : ListImportJobs
	sagesse : ListKnowledgeBases
	sagesse : ListQuickResponses
	sagesse : NotifyRecommendationsReceived
	sagesse : QueryAssistant
	sagesse : RemoveKnowledgeBaseTemplateUri
	sagesse : SearchContent
	sagesse : SearchQuickResponses
	sagesse : SearchSessions
	sagesse : StartContentUpload
	sagesse : StartImportJob
	sagesse : UpdateContent
	sagesse : UpdateKnowledgeBaseTemplateUri
	sagesse : UpdateQuickResponse
	sagesse : UpdateSession

Préfixe de service	Actions
worklink	lien de travail : AssociateDomain lien de travail : AssociateWebsiteAuthorizationProvider lien de travail : AssociateWebsiteCertificateAuthority lien de travail : CreateFleet lien de travail : DeleteFleet lien de travail : DescribeAuditStreamConfiguration lien de travail : DescribeCompanyNetworkConfiguration lien de travail : DescribeDevice lien de travail : DescribeDevicePolicyConfiguration lien de travail : DescribeDomain lien de travail : DescribeFleetMetadata lien de travail : DescribeIdentityProviderConfiguration lien de travail : DescribeWebsiteCertificateAuthority lien de travail : DisassociateDomain lien de travail : DisassociateWebsiteAuthorizationProvider lien de travail : DisassociateWebsiteCertificateAuthority lien de travail : ListDevices lien de travail : ListDomains lien de travail : ListFleets lien de travail : ListWebsiteAuthorizationProviders lien de travail : ListWebsiteCertificateAuthorities

Préfixe de service	Actions
	lien de travail : RestoreDomainAccess lien de travail : RevokeDomainAccess lien de travail : SignOutUser lien de travail : UpdateAuditStreamConfiguration lien de travail : UpdateCompanyNetworkConfiguration lien de travail : UpdateDevicePolicyConfiguration lien de travail : UpdateDomainMetadata lien de travail : UpdateFleetMetadata lien de travail : UpdateIdentityProviderConfiguration

Préfixe de service	Actions
espaces de travail	espaces de travail : AcceptAccountLinkInvitation
	espaces de travail : AssociateConnectionAlias
	espaces de travail : AssociateIpGroups
	espaces de travail : AssociateWorkspaceApplication
	espaces de travail : CopyWorkspaceImage
	espaces de travail : CreateAccountLinkInvitation
	espaces de travail : CreateConnectClientAddIn
	espaces de travail : CreateConnectionAlias
	espaces de travail : CreateIpGroup
	espaces de travail : CreateStandbyWorkspaces
	espaces de travail : CreateUpdatedWorkspaceImage
	espaces de travail : CreateWorkspaceBundle
	espaces de travail : CreateWorkspaceImage
	espaces de travail : CreateWorkspaces
	espaces de travail : CreateWorkspacesPool
	espaces de travail : DeleteAccountLinkInvitation
	espaces de travail : DeleteClientBranding
	espaces de travail : DeleteConnectClientAddIn
	espaces de travail : DeleteConnectionAlias
	espaces de travail : DeleteIpGroup
	espaces de travail : DeleteWorkspaceBundle

Préfixe de service	Actions
	espaces de travail : DeleteWorkspaceImage
	espaces de travail : DeployWorkspaceApplications
	espaces de travail : DeregisterWorkspaceDirectory
	espaces de travail : DescribeAccount
	espaces de travail : DescribeAccountModifications
	espaces de travail : DescribeApplicationAssociations
	espaces de travail : DescribeApplications
	espaces de travail : DescribeBundleAssociations
	espaces de travail : DescribeClientBranding
	espaces de travail : DescribeClientProperties
	espaces de travail : DescribeConnectClientAddIns
	espaces de travail : DescribeConnectionAliasPermissions
	espaces de travail : DescribeConnectionAliases
	espaces de travail : DescribeCustomWorkspaceImageImport
	espaces de travail : DescribeImageAssociations
	espaces de travail : DescribeIpGroups
	espaces de travail : DescribeWorkspaceAssociations
	espaces de travail : DescribeWorkspaceBundles
	espaces de travail : DescribeWorkspaceDirectories
	espaces de travail : DescribeWorkspaceImagePermissions
	espaces de travail : DescribeWorkspaceSnapshots

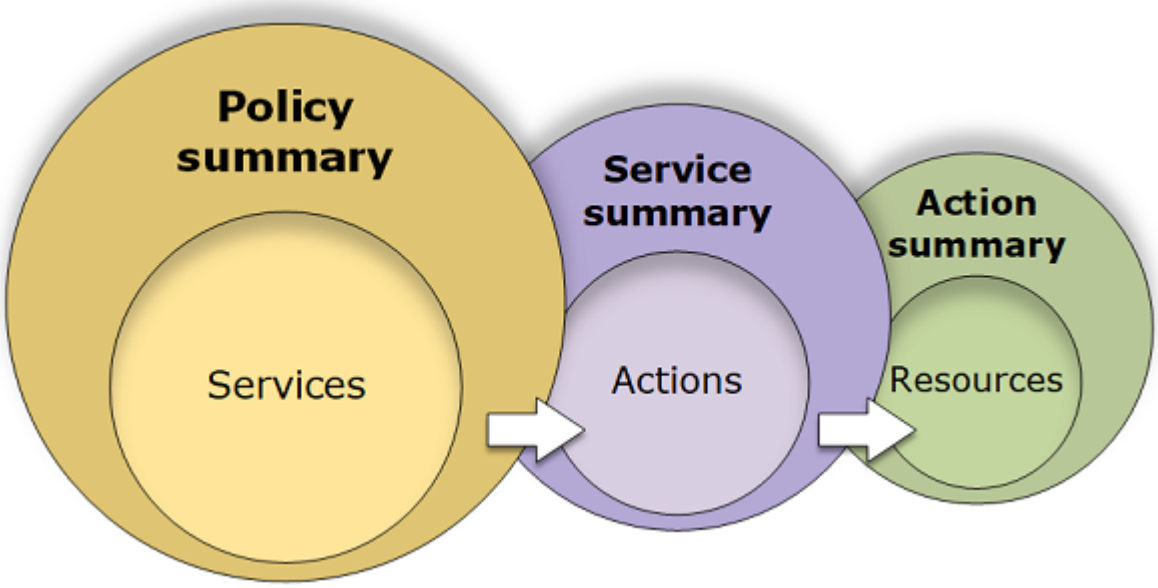
Préfixe de service	Actions
	espaces de travail : DescribeWorkspaces
	espaces de travail : DescribeWorkspacesConnectionStatus
	espaces de travail : DescribeWorkspacesPoolSessions
	espaces de travail : DescribeWorkspacesPools
	espaces de travail : DisassociateConnectionAlias
	espaces de travail : DisassociateIpGroups
	espaces de travail : DisassociateWorkspaceApplication
	espaces de travail : GetAccountLink
	espaces de travail : ImportClientBranding
	espaces de travail : ImportWorkspaceImage
	espaces de travail : ListAccountLinks
	espaces de travail : ListAvailableManagementCidrRanges
	espaces de travail : MigrateWorkspace
	espaces de travail : ModifyAccount
	espaces de travail : ModifyCertificateBasedAuthProperties
	espaces de travail : ModifyClientProperties
	espaces de travail : ModifyEndpointEncryptionMode
	espaces de travail : ModifySamlProperties
	espaces de travail : ModifySelfservicePermissions
	espaces de travail : ModifyStreamingProperties
	espaces de travail : ModifyWorkspaceAccessProperties

Préfixe de service	Actions
	espaces de travail : ModifyWorkspaceCreationProperties
	espaces de travail : ModifyWorkspaceProperties
	espaces de travail : ModifyWorkspaceState
	espaces de travail : RebootWorkspaces
	espaces de travail : RebuildWorkspaces
	espaces de travail : RegisterWorkspaceDirectory
	espaces de travail : RejectAccountLinkInvitation
	espaces de travail : RestoreWorkspace
	espaces de travail : StartWorkspaces
	espaces de travail : StartWorkspacesPool
	espaces de travail : StopWorkspaces
	espaces de travail : StopWorkspacesPool
	espaces de travail : TerminateWorkspaces
	espaces de travail : TerminateWorkspacesPool
	espaces de travail : TerminateWorkspacesPoolSession
	espaces de travail : UpdateConnectClientAddIn
	espaces de travail : UpdateConnectionAliasPermission
	espaces de travail : UpdateWorkspaceBundle
	espaces de travail : UpdateWorkspaceImagePermission
	espaces de travail : UpdateWorkspacesPool

Préfixe de service	Actions
xray	radiographie : CreateGroup radiographie : CreateSamplingRule radiographie : DeleteGroup radiographie : DeleteResourcePolicy radiographie : DeleteSamplingRule radiographie : GetEncryptionConfig radiographie : GetGroup radiographie : GetGroups radiographie : GetInsight radiographie : GetInsightEvents radiographie : GetInsightImpactGraph radiographie : GetInsightSummaries radiographie : GetSamplingRules radiographie : ListResourcePolicies radiographie : PutEncryptionConfig radiographie : PutResourcePolicy radiographie : UpdateGroup radiographie : UpdateSamplingRule

Récapitulatifs de la politique

La console IAM comprend des tables de récapitulatif de la politique qui présentent le niveau d'accès, les ressources et les conditions autorisées ou rejetées pour chaque service dans une politique. Les politiques sont résumées dans trois tables : récapitulatif de la politique, récapitulatif du service et récapitulatif de l'action. La table de récapitulatif de la politique comprend une liste de services. Choisissez un service pour voir le récapitulatif du service. Cette table récapitulative comprend une liste des actions et autorisations associées pour le service choisi. Vous pouvez choisir une action dans cette table pour afficher le récapitulatif de l'action. Cette table comprend une liste des ressources et conditions pour l'action choisie.



Vous pouvez afficher les récapitulatifs de politiques sur la page Utilisateurs ou Rôles pour toutes les politiques (gérées et en ligne) attachées à cet utilisateur. Affichez les récapitulatifs sur la page Politiques pour toutes les politiques gérées. Les politiques gérées incluent les politiques AWS gérées, les politiques AWS gérées par les fonctions de travail et les politiques gérées par le client. Vous pouvez consulter les récapitulatifs de ces politiques sur la page Politiques qu'elles soient attachées à un utilisateur ou à une autre identité IAM.

Vous pouvez utiliser les informations des récapitulatifs de la politique pour comprendre les autorisations accordées ou refusées par votre politique. Les récapitulatifs de la politique peuvent vous aider à résoudre les problèmes des politiques qui ne fournissent pas les autorisations que vous attendiez.

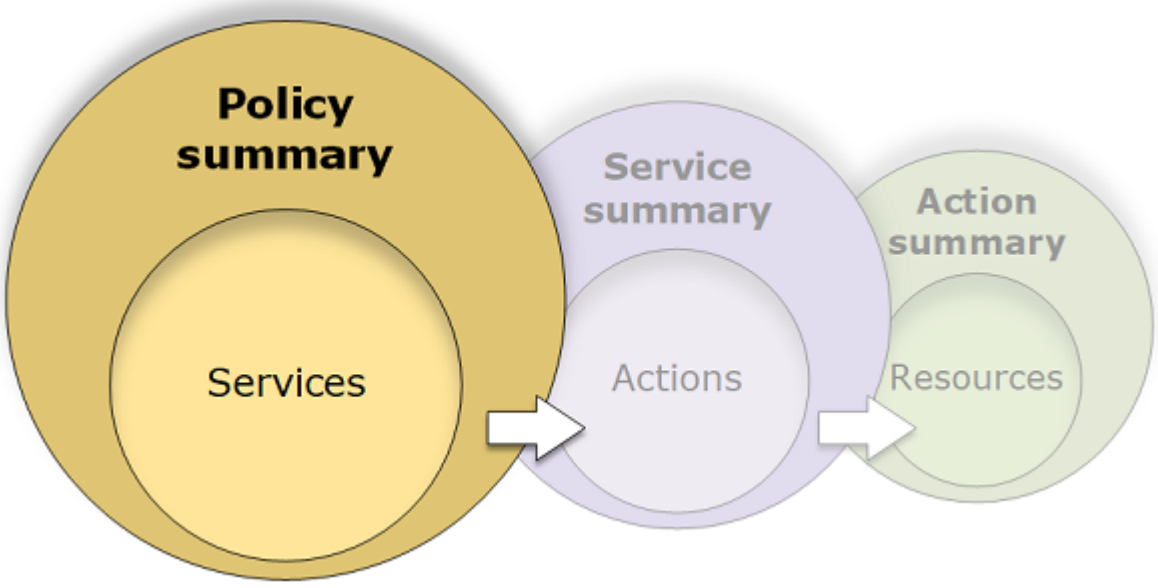
Pour obtenir des instructions détaillées sur l'affichage des résumés, reportez-vous aux Affichage des récapitulatifs de politique sectionsAffichage des récapitulatifs du service, etAffichage des récapitulatifs des actions.

Rubriques

- Récapitulatif de la politique (liste des services)
- Niveaux d'accès dans les récapitulatifs de politique
- Récapitulatif du service (liste des actions)
- Récapitulatif de l'action (liste des ressources)
- Exemples de récapitulatifs de la politique

Récapitulatif de la politique (liste des services)

Les politiques sont résumées dans trois tables : récapitulatif de la politique, récapitulatif du service et récapitulatif de l'action. La table récapitulative de la politique comprend une liste des services et des résumés des autorisations définies par la politique choisie.

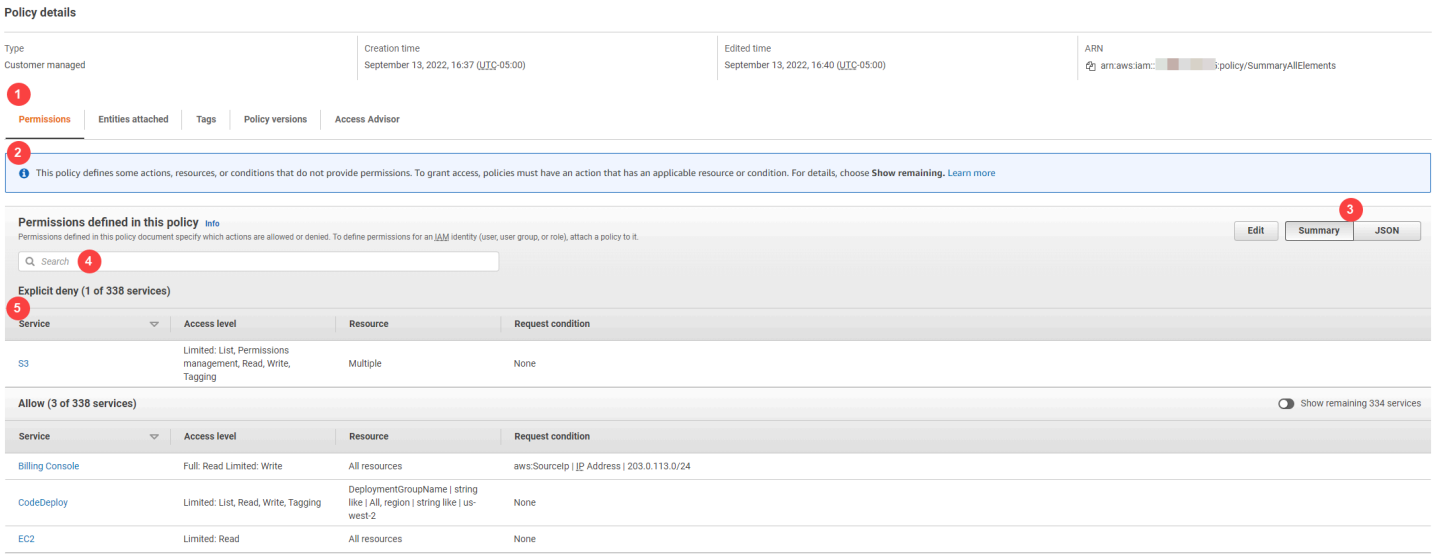


La table récapitulative de la politique est regroupée en une ou plusieurs sections Uncategorized services (Services non catégorisés), Explicit deny (Refus explicite) et Allow (Autoriser). Si la politique comprend un service qu'IAM ne reconnaît pas, le service est inclus dans la section Uncategorized services (Services non catégorisés) de la table. Si IAM reconnaît le service, celui-ci figure dans la

section Explicit deny (Refus explicite) ou Allow (Autoriser) de la table, selon l'effet de la politique (`Deny` ou `Allow`).

Présentation des éléments d'un récapitulatif de service

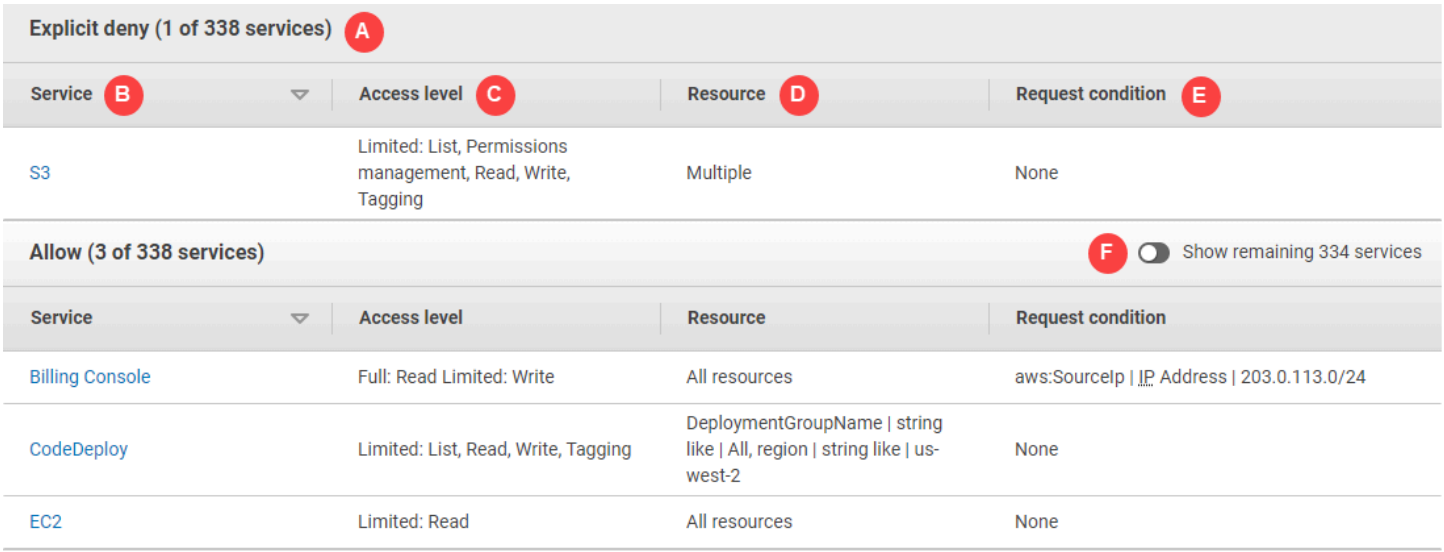
Dans l'exemple suivant de page détaillée d'une politique, la SummaryAllElementspolitique est une politique gérée (politique gérée par le client) attachée directement à l'utilisateur. Cette politique est développée pour afficher son récapitulatif.



Dans l'image précédente, le résumé de la politique est visible sur la page Politiques :

1. L'onglet Autorisations comprend les autorisations définies dans la politique.
2. Si la politique n'accorde pas les autorisations pour toutes les actions, ressources et conditions définies dans la politique, un avertissement ou une bannière d'erreur s'affiche dans la partie supérieure de la page. Puis le récapitulatif de politique inclut des détails relatifs au problème. Pour savoir comment les récapitulatifs de politique vous aident à comprendre les autorisations que votre politique vous accorde et à résoudre les problèmes les concernant, consultez the section called "Ma politique n'accorde pas les autorisations escomptées".
3. Utilisez les boutons Résumé et JSON pour basculer entre le résumé de la politique et le document de politique JSON.
4. Utilisez la zone de Recherche pour réduire la liste des services et trouver un service spécifique.
5. La vue agrandie affiche des détails supplémentaires sur la SummaryAllElementspolitique.

L'image du tableau récapitulatif des politiques ci-dessous montre la SummaryAllElementsstratégie étendue sur la page des détails de la stratégie.



Explicit deny (1 of 338 services) A

Service B	Access level C	Resource D	Request condition E
S3	Limited: List, Permissions management, Read, Write, Tagging	Multiple	None

Allow (3 of 338 services) F Show remaining 334 services

Service	Access level	Resource	Request condition
Billing Console	Full: Read Limited: Write	All resources	aws:SourceIp | IP Address | 203.0.113.0/24
CodeDeploy	Limited: List, Read, Write, Tagging	DeploymentGroupName | string like | All, region | string like | us-west-2	None
EC2	Limited: Read	All resources	None

Dans l'image précédente, le résumé de la politique est visible sur la page Politiques :

A. En ce qui concerne les services qu'IAM reconnaît, il dispose les services selon si la politique autorise ou refuse explicitement l'utilisation du service. Dans cet exemple, la politique inclut une `Deny` déclaration pour le service Amazon S3 et `Allow` des instructions pour les services de facturation et Amazon EC2. CodeDeploy

B. Service : cette colonne répertorie les services définis dans la politique et présente des détails pour chaque service. Chaque nom de service dans la table récapitulative de la politique est un lien vers la table de récapitulatif du service, présenté dans la section Récapitulatif du service (liste des actions). Dans cet exemple, les autorisations sont définies pour les services Amazon S3 CodeDeploy, Billing et Amazon EC2.

C. Niveau d'accès : cette colonne indique si les actions dans chaque niveau d'accès (`List`, `Read`, `Write`, `Permission Management` et `Tagging`) ont des autorisations `Full` ou `Limited` définies dans la politique. Pour plus d'informations et pour consulter des exemples de récapitulatif de niveau d'accès, consultez Niveaux d'accès dans les récapitulatifs de politique.

- Full access (Accès complet) : cette entrée indique que le service a accès à toutes les actions dans les quatre niveaux d'accès disponibles pour le service.
- Si l'entrée ne comprend pas l'Accès complet, le service a accès à certains actions, mais pas à l'ensemble des actions, pour le service. L'accès est alors défini en suivant les descriptions pour chacune des classifications du niveau d'accès (`List`, `Read`, `Write`, `Permission Management` et `Tagging`) :

Complet : La politique fournit un accès complet à toutes les actions contenues dans chaque classification de niveau d'accès répertoriée. Dans cet exemple, la politique permet l'accès à toutes les actions `Read` de la facturation.

Limité : La politique fournit un accès à une ou plusieurs actions, mais pas à l'ensemble des actions contenues dans chaque classification de niveau d'accès répertoriée. Dans cet exemple, la politique permet l'accès à certaines des actions `Write` de la facturation.

D. Resource (Ressource) : cette colonne présente les ressources que la politique définit pour chaque service.

- Multiple (Plusieurs) : la politique comprend plusieurs ressources, mails pas leur totalité, dans le service. Dans cet exemple, l'accès est explicitement refusé à plusieurs ressources Amazon S3.
- Toutes les ressources : la stratégie est définie pour toutes les ressources du service. Dans cet exemple, la politique autorise les actions répertoriées sur toutes les ressources de la facturation.
- Resource text (Texte de ressource) : la politique contient une ressource dans le service. Dans cet exemple, les actions répertoriées ne sont autorisées que sur la `DeploymentGroupName` CodeDeploy ressource. En fonction des informations que le service fournit à IAM, il est possible que vous voyiez un ARN ou le type de ressource défini.

> **Note**
>
> Cette colonne peut contenir une ressource provenant d'un service différent. Si l'instruction de politique qui inclut la ressource ne contient pas d'actions et de ressources provenant d'un même service, cela signifie que votre politique contient des ressources non appariées. IAM ne vous avertit pas que des ressources ne sont pas appariées lorsque vous créez une politique ou que vous affichez une politique dans le récapitulatif de la politique. Si cette colonne contient une ressource non appariée, recherchez d'éventuelles erreurs dans votre politique. Afin de mieux comprendre vos politiques, vous devez toujours les tester avec le simulateur de politique.

E. Request condition (Demander une condition) : cette colonne indique si les services ou actions associés aux ressources font l'objet de conditions.

- None (Aucune) : la politique ne contient aucune condition pour le service. Dans cet exemple, aucun condition n'est appliquée aux actions refusées dans le service Amazon S3.

- Condition text (Texte de condition) : la politique contient une condition pour le service. Dans cet exemple, les actions de Facturation énumérées sont autorisées uniquement si l'adresse IP de la source correspond à `203.0.113.0/24`.
- Multiple (Plusieurs) : la politique contient plusieurs conditions pour le service. Pour afficher chacune des conditions multiples de la politique, choisissez JSON pour afficher le document de politique.

F. Afficher les services restants : activez ce bouton pour développer la table afin d'inclure les services non définis par la politique. Ces services sont refusés implicitement (ou refusés par défaut) dans cette politique. Cependant, une instruction dans une autre politique peut tout de même autoriser ou refuser explicitement l'utilisation du service. Le récapitulatif de la politique résume les autorisations d'une seule politique. Pour savoir comment le AWS service décide si une demande donnée doit être autorisée ou refusée, voirLogique d'évaluation de politiques.

Lorsqu'une politique ou un élément dans la politique n'accorde pas d'autorisations, IAM fournit des avertissements et des informations supplémentaires dans le récapitulatif de politique. Le tableau récapitulatif de la politique suivant montre les services étendus Afficher les services restants sur la page des détails de la SummaryAllElementspolitique avec les avertissements possibles.

Explicit deny (1 of 338 services)

Service	Access level	Resource a	Request condition b
S3	Limited: List, Permissions management, Read, Write, Tagging	c Multiple| ⚠ One or more actions do not have an applicable resource.	None

Allow (3 of 338 services) Show remaining 334 services

Service	Access level	Resource	Request condition
Billing Console	Full: Read Limited: Write	All resources	aws:SourceIp | IP Address | 203.0.113.0/24
CodeCommit	None	d ⚠ No resources are defined.	None
CodeDeploy	Limited: List, Read, Write, Tagging	e DeploymentGroupName | string like | All, region | string like | us-west-2| ⚠ One or more actions do not have an applicable resource.	None
EC2	Limited: Read	All resources	None
S3	None	None| ⚠ One or more actions do not have an applicable resource.	f None | ⚠ One or more conditions do not have an applicable action.

Dans l'image précédente, vous pouvez voir tous les services incluant des actions, ressources ou conditions définies sans autorisation :

a. Resource warnings (Avertissements relatifs aux ressources) : pour les services qui ne fournissent pas d'autorisations pour toutes les actions ou ressources incluses, vous voyez l'un des avertissements suivants dans la colonne Resource (Ressource) de la table :

   - 

     Aucune ressource n'est définie. – Cela signifie que le service a défini des actions, mais qu'aucune ressource prise en charge n'est incluse dans la politique.

   - 

     Une ou plusieurs actions n'ont pas de ressource applicable. – Cela signifie que le service a défini des actions, mais que certaines d'entre elles n'ont pas de ressource prise en charge.

   - 

     Une ou plusieurs ressources n'ont pas d'action applicable. – Cela signifie que le service a défini des ressources, mais que certaines d'entre elles n'ont pas d'action associée.

   Si un service comprend à la fois des actions qui n'ont pas de ressource applicable et des ressources qui ont une ressource applicable, alors seul l'avertissement Une ou plusieurs ressources n'ont pas d'action applicable est affiché. Ceci est dû au fait que lorsque vous affichez le récapitulatif de ce service, les ressources qui ne s'appliquent à aucune action ne s'affichent pas. Pour l'action `ListAllMyBuckets`, cette politique inclut le dernier avertissement, car l'action ne prend pas en charge les autorisations au niveau des ressources, ni la clé de condition `s3:x-amz-acl`. Si vous corrigez le problème de ressource ou celui de condition, le problème restant s'affiche dans un avertissement détaillé.

b. Request condition warnings (Avertissements relatifs aux demandes de condition) : pour les services qui ne fournissent pas d'autorisations pour toutes les conditions incluses, vous voyez l'un des avertissements suivants dans la colonne Request condition (Demander une condition) de la table :

   - 

     Une ou plusieurs actions n'ont pas de condition applicable. – Cela signifie que le service a défini des actions, mais que certaines d'entre elles n'ont pas de condition prise en charge.

   - 

     Une ou plusieurs conditions n'ont pas d'action applicable. – Cela signifie que le service a défini des conditions, mais que certaines d'entre elles n'ont pas d'action associée.

c. Multiple (Plusieurs) |

Une ou plusieurs actions n'ont pas de ressource applicable. – L'instruction `Deny` pour Amazon S3 inclut plusieurs ressources. Elle inclut également plusieurs actions ; certaines actions prennent en charge les ressources et d'autres non. Pour afficher cette politique, consultez the section called "SummaryAllElements Document de politique JSON". Dans ce cas, la politique inclut toutes les actions d'Amazon S3, et seules les actions qui peuvent être effectuées sur un compartiment ou un objet dans un compartiment sont rejetées.

d. ⚠

No resources are defined (Aucune ressource n'est définie) : le service a défini des actions, mais aucune ressource prise en charge n'est incluse dans la politique, et par conséquent, le service ne fournit aucune autorisation. Dans ce cas, la politique inclut des CodeCommit actions mais aucune CodeCommit ressource.

e. DeploymentGroupName | string like | All, region | string like | us-west-2

⚠

| Aucune ressource applicable n'est associée à une ou plusieurs actions. – Le service a une action définie et au moins une autre action qui n'a pas de ressource de support.

f. None |

⚠

Une ou plusieurs conditions n'ont pas d'action applicable. – Le service a au moins une clé de condition qui n'a pas d'action de support.

SummaryAllElements Document de politique JSON

La SummaryAllElementspolitique n'est pas destinée à être utilisée pour définir des autorisations dans votre compte. En fait, elle est incluse pour illustrer les erreurs et les avertissements que vous risquez de rencontrer lors de l'affichage d'un récapitulatif de politique.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "billing:Get*",
                "payments:List*",
```

```
                "payments:Update*",
                "account:Get*",
                "account:List*",
                "cur:GetUsage*"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action": [
                "s3:*"
            ],
            "Resource": [
                "arn:aws:s3:::customer",
                "arn:aws:s3:::customer/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshots"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "codedploy:*",
                "codecommit:*"
            ],
            "Resource": [
                "arn:aws:codedeploy:us-west-2:123456789012:deploymentgroup:*",
                "arn:aws:codebuild:us-east-1:123456789012:project/my-demo-
project"
            ]
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets",
                "s3:GetObject",
                "s3:DeletObject",
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*",
                "arn:aws:autoscling:us-east-2:123456789012:autoscalgrp"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": [
                        "public-read"
                    ],
                    "s3:prefix": [
                        "custom",
                        "other"
                    ]
                }
            }
        }
    ]
}
```

Affichage des récapitulatifs de politique

Vous pouvez afficher le récapitulatif de politique pour des politiques attachées à un rôle ou à un utilisateur IAM. Pour les politiques gérées, vous pouvez afficher le récapitulatif des politiques sur la page Politiques. Si votre politique ne comprend pas de récapitulatif de la politique, consultez Récapitulatif de politique manquant pour en comprendre la raison.

Affichage des récapitulatifs de politique sur la page Politiques

Vous pouvez afficher le récapitulatif des politiques gérées sur la page Politiques.

Pour afficher le récapitulatif de la stratégie sur la page Stratégies

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de la stratégie à afficher.
4. Sur la page Détails de la politique, affichez l'onglet Autorisations pour consulter le résumé de la politique.

Affichage du récapitulatif d’une politique attachée à un utilisateur

Vous pouvez afficher le récapitulatif de politique pour toute politique attachée à un utilisateur IAM.

Pour afficher le récapitulatif d'une politique attachée à un utilisateur

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Choisissez Utilisateurs dans le panneau de navigation.
3. Dans la liste des utilisateurs, choisissez le nom de l'utilisateur dont vous souhaitez afficher la politique.
4. Sur la page Récapitulatif de l'utilisateur, affichez l'onglet Autorisations pour afficher la liste des stratégies attachées à l'utilisateur directement ou via un groupe.
5. Dans le tableau des stratégies dédiées à l'utilisateur, développez la ligne de la stratégie à afficher.

Affichage du récapitulatif d’une politique attachée à un rôle

Vous pouvez afficher le récapitulatif de politique pour toute politique attachée à un rôle.

Pour afficher le récapitulatif d'une politique attachée à un rôle

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Rôles.
3. Dans la liste des rôles, choisissez le nom du rôle dont vous souhaitez afficher la politique.

4. Sur la page Récapitulatif du rôle, affichez l'onglet Autorisations pour afficher la liste des stratégies attachées au rôle.
5. Dans le tableau des stratégies dédiées au rôle, développez la ligne de la stratégie à afficher.

Modification des politiques pour corriger les avertissements

Lorsque vous consultez le récapitulatif d'une politique, vous pouvez détecter une faute de frappe ou remarquer que la politique ne fournit pas les autorisations que vous attendiez. Vous ne pouvez pas modifier le récapitulatif d'une politique directement. Toutefois, vous pouvez modifier une politique gérée par le client à l'aide de l'éditeur de politique visuel, qui détecte un grand nombre des mêmes erreurs et avertissements que les rapports de synthèse de politique. Vous pouvez ensuite consulter les modifications dans le résumé de politique pour confirmer que vous avez résolu tous les problèmes. Pour savoir comment modifier une politique en ligne, consulter the section called “Modification de politiques IAM”. Vous ne pouvez pas modifier les politiques AWS gérées.

Vous pouvez modifier une politique pour votre récapitulatif de la politique en utilisant l’option Visuel.

Pour modifier une politique pour votre résumé de la politique à l'aide de l'option Visuel

1. Ouvrez le récapitulatif de politique comme expliqué dans les procédures précédentes.
2. Choisissez Modifier.

   Si vous vous trouvez sur la page Utilisateurs et que vous choisissez de modifier une stratégie gérée par le client qui est attachée à ce dernier, vous êtes redirigé vers la page Stratégies. Vous ne pouvez modifier les stratégies gérées par le client que sur la page Stratégies.
3. Choisissez l'option Visuel pour afficher la représentation visuelle modifiable de votre politique. IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel et de vous permettre de trouver et de résoudre plus facilement tous les problèmes. Les avertissements et messages d'erreur sur la page peuvent vous aider à résoudre tous les problèmes liés à votre politique. Pour plus d'informations sur les politiques de restructuration IAM, voir Restructuration de politique.
4. Modifiez votre politique et choisissez Suivant pour que vos modifications s'affichent dans le résumé de la politique. Si un problème persiste, choisissez Précédent pour retourner à l'écran de modification.
5. Choisissez Enregistrer les Modifications pour enregistrer vos Modifications.

Vous pouvez modifier une politique pour votre récapitulatif de la politique en utilisant l’option JSON.

Pour modifier une politique pour votre récapitulatif de politique à l'aide de l'option JSON

1. Ouvrez le récapitulatif de politique comme expliqué dans les procédures précédentes.
2. Vous pouvez utiliser les boutons Résumé et JSON pour comparer le résumé de la politique au document de politique JSON. Ces informations peuvent vous permettre de déterminer quelles lignes du document de politique vous souhaitez modifier.
3. Choisissez Modifier, puis l'option JSON pour modifier le document de politique JSON.

   > **Note**
   >
   > Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'option éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour de plus amples informations, veuillez consulter Restructuration de politique.

   Si vous vous trouvez sur la page Utilisateurs et que vous choisissez de modifier une stratégie gérée par le client qui est attachée à ce dernier, vous êtes redirigé vers la page Stratégies. Vous ne pouvez modifier les stratégies gérées par le client que sur la page Stratégies.
4. Modifiez votre politique. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant. Si un problème persiste, choisissez Précédent pour retourner à l'écran de modification.
5. Choisissez Enregistrer les Modifications pour enregistrer vos Modifications.

Niveaux d'accès dans les récapitulatifs de politique

AWS résumé des niveaux d'accès

Les récapitulatifs de politique comprennent un récapitulatif de niveau d'accès décrivant les autorisations d'action définies pour chaque service mentionné dans la politique. Pour en savoir plus sur les récapitulatifs de politiques, consultez Récapitulatifs de la politique. Les récapitulatifs de niveau d'accès indiquent si les actions dans chaque niveau d'accès (`List`, `Read`, `Tagging`, `Write`, et `Permissions management`) ont des autorisations `Full` ou `Limited` définies dans la politique. Pour consulter la classification des niveaux d'accès attribuée à chaque action d'un service, consultez la section Actions, ressources et clés de condition pour les AWS services.

L'exemple suivant illustre l'accès fourni par une politique pour les services donnés. Pour consulter des exemples de documents de politique JSON complets et les récapitulatifs associés, consultez Exemples de récapitulatifs de la politique.

Service	Niveau d'accès	Cette politique fournit les autorisations suivantes :
IAM	Accès complet à	Accès à toutes les actions au sein du service IAM.
CloudWatch	Complet : List (Liste)	Accès à toutes les CloudWatch actions du niveau `List` d'accès, mais pas d'accès aux actions correspondant à la classification `ReadWrite`, ou au niveau `Permissions management` d'accès.
Data Pipeline	Limité : List (Liste), Read (Lire)	Accès à au moins une des AWS Data Pipeline actions du niveau d'`Read`accès `List` et, mais pas à toutes, mais pas aux `Permissions management` actions `Write` ou.
EC2	Complet : List (Liste), Read (Lire) Limité : Write (Écrire)	Accès à toutes les actions Amazon EC2 `List` et `Read`, et accès à au moins une action Amazon EC2 `Write`, mais à pas toutes, mais aucun accès aux actions ayant la classific ation de niveau d'accès `Permissions management` .
S3	Limité : Read (Lire), Write (Écrire), Permissions management (Gestion des autorisations)	Accès à au moins une action Amazon S3 `Read`, `Write` et `Permissions management` , mais pas toutes à la fois.
codedploy	(vide)	Accès inconnu, car IAM ne reconnaît pas ce service.
API Gateway	Aucune	Aucun accès n'est défini dans la politique.

Service	Niveau d'accès	Cette politique fournit les autorisations suivantes :
CodeBuild	⚠ Aucune action n'est définie.	Aucun accès, car aucune action n'est définie pour le service. Pour comprendre et corriger ce problème, veuillez consulter the section called "Ma politique n'accorde pas les autorisations escomptées".

Dans le récapitulatif d'une politique, Accès complet indique que la politique donne accès à toutes les actions au sein du service. Les politiques permettant l'accès à certaines, mais pas à l'ensemble des actions contenues dans un service sont regroupées en fonction de la classification de niveau d'accès. Cela est indiqué par un des regroupements de niveaux d'accès suivants :

- Complet : La politique fournit un accès complet à toutes les actions contenues dans la classification de niveau d'accès spécifiée.
- Limité : La politique fournit un accès à une ou plusieurs actions, mais pas à l'ensemble des actions contenues dans la classification de niveau d'accès spécifiée.
- Aucun : La politique ne fournit aucun accès.
- (vide) : IAM ne reconnaît pas ce service. Si le nom du service comprend une faute de frappe, la politique ne fournit aucun accès au service. Si le nom du service est correct, il se peut que le service ne prenne pas en charge les récapitulatifs de politique ou qu'il soit en mode aperçu. Dans ce cas, la politique peut accorder l'accès, mais celui-ci ne peut pas être affiché dans le récapitulatif de la politique. Pour demander la prise en charge du récapitulatif de politique d'un service disponible pour tous (GA), consultez Le service ne prend pas en charge les récapitulatifs de politique IAM.

Les résumés des niveaux d'accès qui incluent un accès limité (partiel) aux actions sont regroupés à l'aide des classifications des niveaux d' AWS accès `ListRead`, `TaggingWrite`, ou`Permissions management`.

AWS niveaux d'accès

AWS définit les classifications de niveau d'accès suivantes pour les actions d'un service :

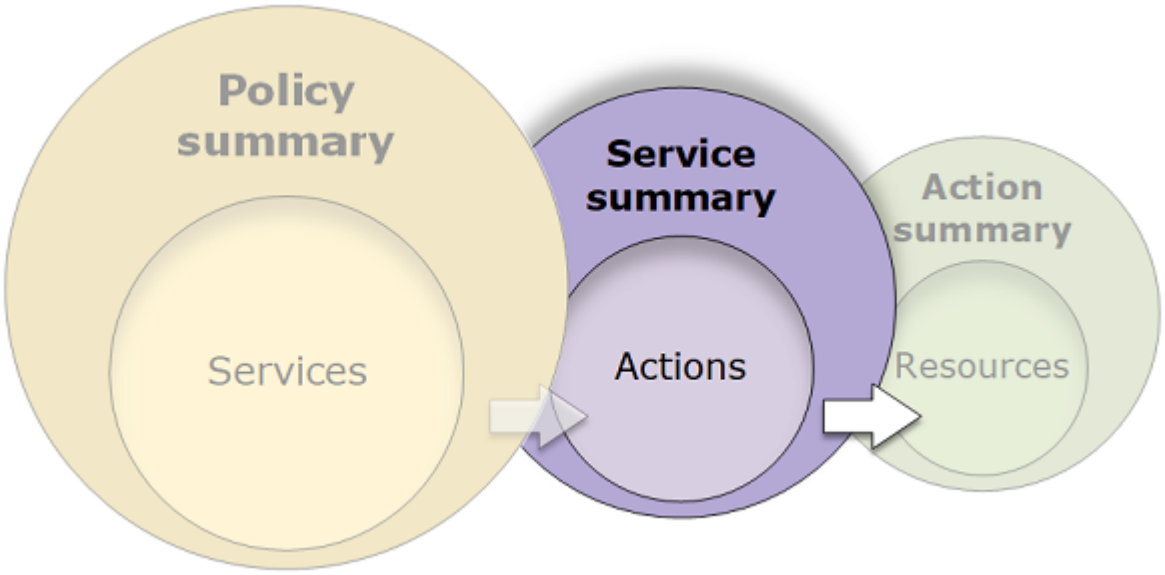
- List (Liste) : Autorisation de répertorier les ressources au sein du service afin de déterminer si un objet existe. Les actions associées à ce niveau d'accès peuvent répertorier les objets mais ne peuvent pas voir le contenu d'une ressource. Par exemple, l'action `ListBucket` Amazon S3 possède le niveau d'accès List (Liste).
- Read (Lire) : Autorisation de lire le contenu et les attributs de ressources dans le service, mais pas de les modifier. Par exemple, les actions Amazon S3 `GetObject` et `GetBucketLocation` possèdent le niveau d'accès Read (Lecture).
- Balisage : Autorisation d'effectuer des actions qui modifient uniquement l'état des balises de ressource. Par exemple, les actions IAM `TagRole` et `UntagRole` ont le niveau d'accès Tagging (Balisage), car elles autorisent uniquement le balisage ou l'annulation du balisage d'un rôle. Cependant, l'action `CreateRole` autorise le balisage d'une ressource de rôle lorsque vous créez ce rôle. Étant donné que l'action n'ajoute pas uniquement une balise, elle possède le niveau d'accès `Write`.
- Write (Écrire) : autorisation de créer, supprimer ou modifier des ressources du service. Par exemple, les actions Amazon S3 `CreateBucket` `DeleteBucket` et `PutObject` possèdent le niveau d'accès Write (Écriture). Les actions `Write` peuvent également autoriser la modification d'une balise de ressource. Toutefois, une action qui autorise uniquement les modifications des balises possède le niveau d'accès `Tagging`.
- Gestion des autorisations : la gestion des autorisations fait référence aux actions qui contrôlent l'accès interne Services AWS, y compris les autorisations d'identité IAM et non-IAM, mais exclut les contrôles d'accès au niveau du réseau tels que les groupes de sécurité. Par exemple, la plupart des AWS Organizations actions IAM et, ainsi que les actions Amazon S3, `DeleteBucketPolicy` disposent du niveau `PutBucketPolicy` d'accès de gestion des autorisations.

> **Conseil**
>
> Pour améliorer la sécurité de vos politiques Compte AWS, restreignez ou surveillez régulièrement les politiques qui incluent la classification des niveaux d'accès de la gestion des autorisations.

Pour afficher la classification des niveaux d'accès pour toutes les actions d'un service, voir Actions, ressources et clés de condition pour les AWS services.

Récapitulatif du service (liste des actions)

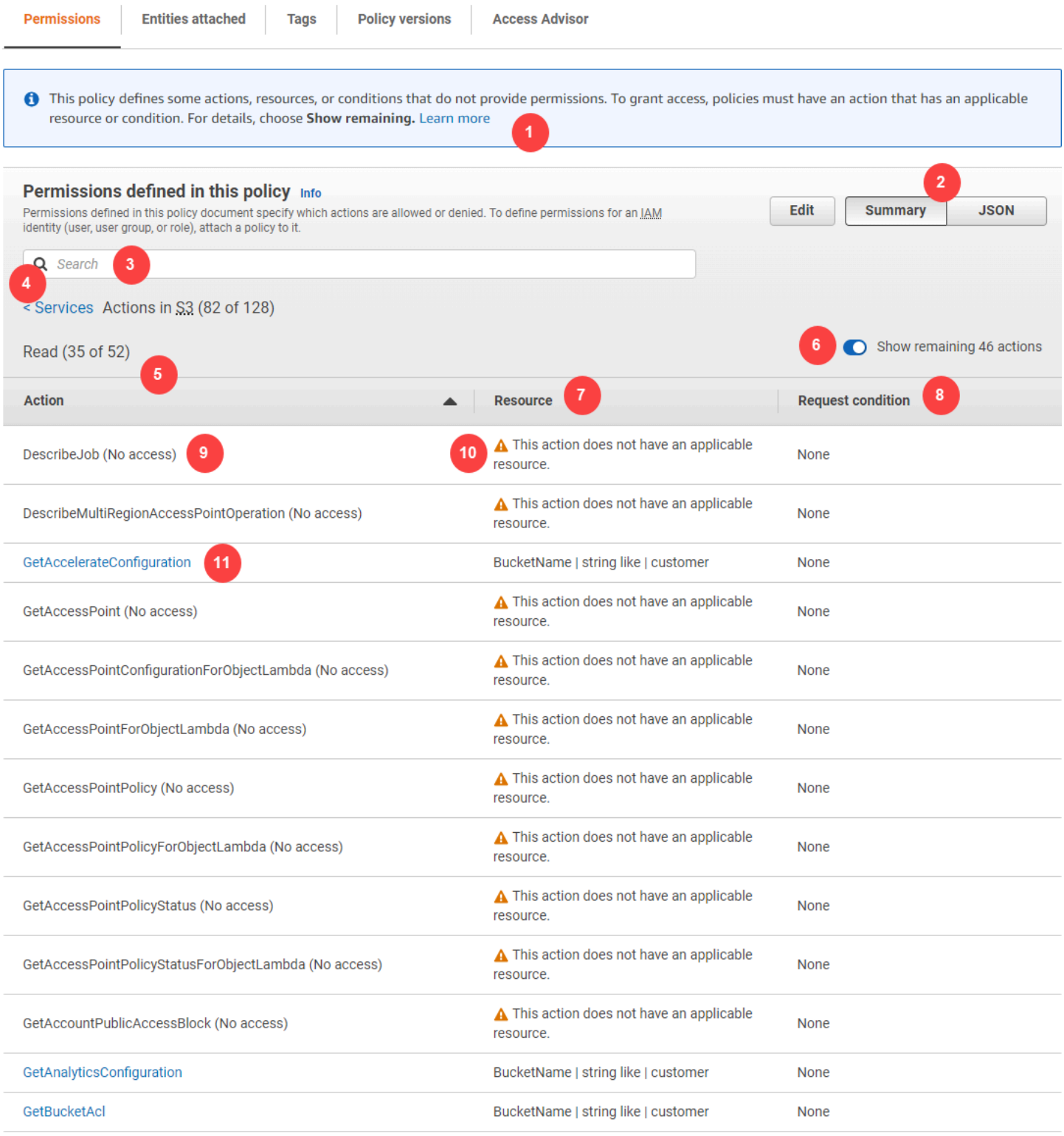
Les politiques sont résumées dans trois tables : récapitulatif de la politique, récapitulatif du service et récapitulatif de l'action. La table du récapitulatif du service inclut une liste des actions et des récapitulatifs des autorisations définies par la politique pour le service choisi.



Vous pouvez consulter un récapitulatif du service pour chaque service répertorié dans le récapitulatif de la politique qui accorde les autorisations. La table est regroupée en sections Uncategorized actions (Actions non catégorisées), Uncategorized resource types (Types de ressources non catégorisées) et de niveau d'accès. Si la politique comprend une action qu'IAM ne reconnaît pas, l'action est incluse dans la section Uncategorized actions (Actions non catégorisées) de la table. Si IAM reconnaît l'action, elle est alors incluse dans l'une des sections de niveau d'accès (liste, lecture, écriture et gestion des autorisations) du tableau. Pour consulter la classification des niveaux d'accès attribuée à chaque action d'un service, consultez la section Actions, ressources et clés de condition pour les AWS services.

Présentation des éléments d'un récapitulatif du service

L'exemple suivant représente le résumé des services pour les actions Amazon S3 qui sont autorisées à partir d'un résumé de la politique. Les actions de ce service sont groupées par niveau d'accès. Par exemple, 35 actions Lire sont définies sur un total de 52 actions Lire disponibles pour le service.



La page de récapitulatif du service pour une politique gérée inclut les informations suivantes :

1. Si la politique n'accorde pas les autorisations pour toutes les actions, ressources et conditions définies pour le service dans la politique, un avertissement ou une bannière d'erreur s'affiche

dans la partie supérieure de la page. Puis le récapitulatif du service inclut des détails relatifs au problème. Pour savoir comment les récapitulatifs de politique vous aident à comprendre les autorisations que votre politique vous accorde et à résoudre les problèmes les concernant, consultez the section called "Ma politique n'accorde pas les autorisations escomptées".

2. Choisissez JSON pour consulter des détails supplémentaires sur la politique. Cela peut vous servir pour afficher l'ensemble des conditions appliquées aux actions. (Si vous consultez le récapitulatif du service pour une politique en ligne attachée directement à un utilisateur, vous devez fermer la boîte de dialogue du récapitulatif du service et revenir au récapitulatif de la politique pour accéder au document de politique JSON.)
3. Pour consulter le résumé d'une action spécifique, tapez des mots-clés dans la zone de Recherche afin de réduire la liste des actions disponibles.
4. En regard de la flèche de retour Services apparaît le nom du service (dans ce cas S3). Le résumé des services pour ce service inclut la liste des actions autorisées ou refusées définies dans la politique. Si le service apparaît sous (Refus explicite) dans l'onglet Autorisations, alors les actions énumérées dans la table récapitulative des services sont explicitement refusées. Si le service apparaît sous Autoriser dans l'onglet Autorisations, alors les actions énumérées dans la table récapitulative des services sont autorisées.
5. Action : cette colonne énumère les actions définies dans la politique et fournit les ressources et les conditions pour chaque action. Si la politique accorde ou refuse des autorisations à l'action, alors le nom de l'action renvoie à la table récapitulative de l'action. La table regroupe ces actions dans au moins une ou jusqu'à cinq sections au maximum, en fonction du niveau d'accès autorisé ou refusé par la politique. Les sections sont Liste, Lire, Écrire, Gestion des autorisations et Balisage. Le compte indique le nombre d'actions reconnues qui fournissent des autorisations pour chaque niveau d'accès. Le total correspond au nombre d'actions connues pour le service. Dans cet exemple, 35 actions fournissent des autorisations sur un total de 52 actions connues Amazon S3 Lire. Pour consulter la classification des niveaux d'accès attribuée à chaque action d'un service, consultez la section Actions, ressources et clés de condition pour les AWS services.
6. Afficher les actions restantes : activez ce bouton pour développer ou masquer la table afin d'inclure des actions connues mais qui ne fournissent pas d'autorisations pour ce service. L'activation du bouton permet également d'afficher des avertissements pour tous les éléments qui ne fournissent pas d'autorisations.
7. Resource (Ressource) : cette colonne présente les ressources que la politique définit pour le service. IAM ne vérifie pas si la ressource s'applique à chaque action. Dans cet exemple, les actions du service Amazon S3 sont autorisées uniquement sur la ressource du compartiment Amazon S3 `developer_bucket`. En fonction des informations que le service fournit à IAM,

vous verrez s'afficher un ARN, tel que `arn:aws:s3:::developer_bucket/*`, ou le type de ressource défini, tel que `BucketName = developer_bucket`.

> **Note**
>
> Cette colonne peut contenir une ressource provenant d'un service différent. Si l'instruction de politique qui inclut la ressource ne contient pas d'actions et de ressources provenant d'un même service, cela signifie que votre politique contient des ressources non appariées. IAM ne vous avertit pas que des ressources ne sont pas appariées lorsque vous créez une politique ou que vous affichez une politique dans le récapitulatif du service. IAM n'indique pas non plus si l'action s'applique aux ressources, uniquement si le service correspond. Si cette colonne contient une ressource non appariée, recherchez d'éventuelles erreurs dans votre politique. Afin de mieux comprendre vos politiques, vous devez toujours les tester avec le simulateur de politique.

8. Request condition (Demander une condition) : cette colonne indique si les services ou actions associés aux ressources font l'objet de conditions. Pour en savoir plus sur ces conditions, choisissez JSON pour examiner le document de politique JSON.
9. (No access) (Pas d'accès) : cette politique inclut une action qui ne fournit pas d'autorisations.
10. Resource warning (Avertissement concernant les ressources) : pour les actions dont les ressources ne fournissent pas d'autorisations complètes, l'un des avertissements suivants s'affiche :
    - Cette action ne prend pas en charge les autorisations au niveau des ressources. Cet élément exige un caractère générique (*) pour la ressource. – Cela signifie que la politique inclut des autorisations au niveau des ressources, mais qu'elle doit inclure `"Resource": ["*"]` pour fournir des autorisations pour cette action.
    - Cette action n'a pas de ressource applicable. – Cela signifie que l'action est incluse dans la politique sans ressource prise en charge.
    - Cette action n'a pas de ressource et de condition applicable. – Cela signifie que l'action est incluse dans la politique sans ressource prise en charge, ni condition prise en charge. Dans ce cas, il y a également une condition incluse dans la politique pour ce service, mais il n'existe pas de conditions qui s'appliquent à cette action.
11. Les actions qui fournissent des autorisations incluent un lien vers le récapitulatif de l'action.

Affichage des récapitulatifs du service

Vous pouvez consulter un récapitulatif du service pour chaque service répertorié dans le récapitulatif de la politique qui accorde les autorisations.

Affichage des récapitulatifs du service à partir de la page Politiques

Vous pouvez afficher le résumé des services pour les politiques gérées sur la page Politiques.

Pour afficher le récapitulatif du service d'une politique gérée

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de la stratégie à afficher.
4. Sur la page Détails de la politique, affichez l'onglet Autorisations pour consulter le résumé de la politique.
5. Dans la liste des services du récapitulatif de la politique, choisissez le nom du service à afficher.

Affichage du récapitulatif du service d'une politique attachée à un utilisateur

Vous pouvez consulter le récapitulatif des services pour toutes les politiques attachées à un utilisateur IAM.

Pour afficher le récapitulatif du service d'une politique attachée à un utilisateur

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Dans la liste des utilisateurs, choisissez le nom de l'utilisateur dont vous souhaitez afficher la politique.
4. Sur la page Récapitulatif de l'utilisateur, affichez l'onglet Autorisations pour afficher la liste des stratégies attachées à l'utilisateur directement ou via un groupe.
5. Dans la table des politiques de l'utilisateur, choisissez le nom de la politique que vous voulez afficher.

Si vous vous trouvez sur la page Utilisateurs et vous choisissez d'afficher le résumé des services pour une politique attachée à cet utilisateur, vous êtes redirigé vers la page Politiques. Vous pouvez afficher les résumés des services uniquement sur la page Politiques.

6. Choisissez Résumé. Dans la liste des services du récapitulatif de la politique, choisissez le nom du service à afficher.

> **Note**
>
> Si la politique que vous sélectionnez est une politique en ligne attachée directement à l'utilisateur, le tableau récapitulatif du service s'affiche. Si la politique est une politique en ligne attachée via un groupe, vous êtes dirigé vers le document de politique JSON de ce groupe. Si la stratégie est une stratégie gérée, vous êtes dirigé vers le récapitulatif du service de cette stratégie sur la page Stratégies.

Affichage du récapitulatif du service d'une politique attachée à un rôle

Vous pouvez afficher le récapitulatif de politique pour toute politique attachée à un rôle.

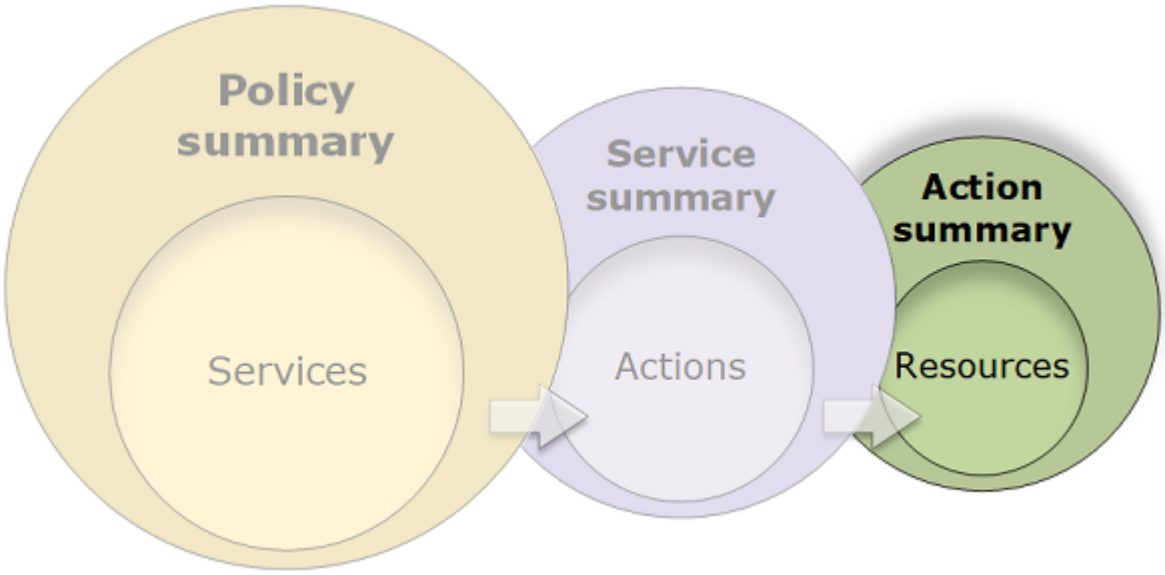
Pour afficher le récapitulatif du service d'une politique attachée à un rôle

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Choisissez Rôles dans le panneau de navigation.
3. Dans la liste des rôles, choisissez le nom du rôle dont vous souhaitez afficher la politique.
4. Sur la page Récapitulatif du rôle, affichez l'onglet Autorisations pour afficher la liste des stratégies attachées au rôle.
5. Dans la table des politiques pour le rôle, choisissez le nom de la politique que vous voulez afficher.

   Si vous vous trouvez sur la page Rôles et vous choisissez d'afficher le résumé des services pour une politique attachée à cet utilisateur, vous êtes redirigé vers la page Politiques. Vous pouvez afficher les résumés des services uniquement sur la page Politiques.
6. Dans la liste des services du récapitulatif de la politique, choisissez le nom du service à afficher.

Récapitulatif de l'action (liste des ressources)

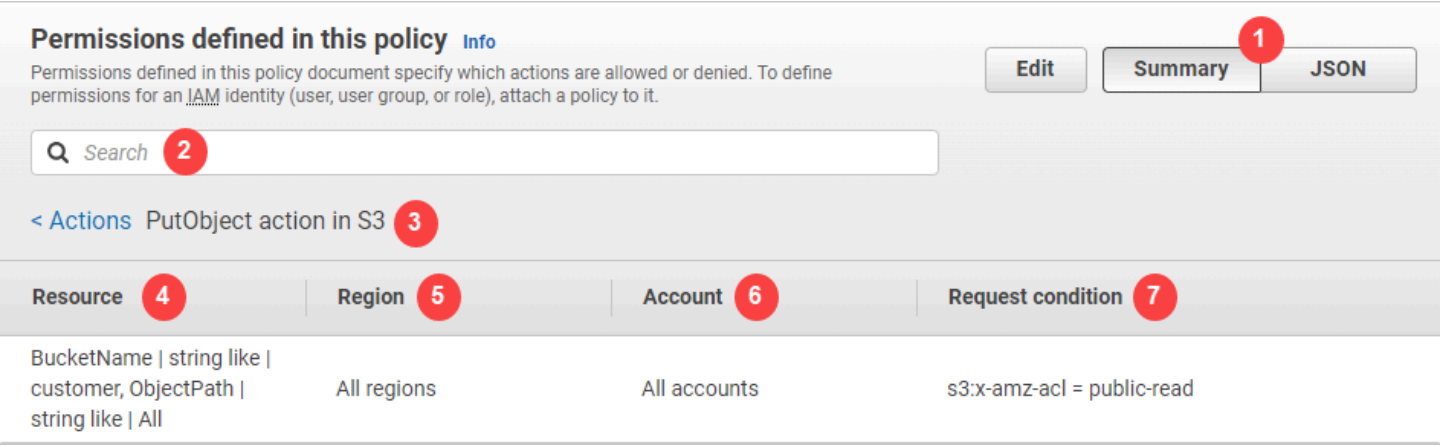
Les politiques sont résumées dans trois tables : récapitulatif de la politique, récapitulatif du service et récapitulatif de l'action. La table récapitulative de l'action inclut une liste des ressources et des conditions associées qui s'appliquent à l'action choisie.



Pour afficher un récapitulatif de l'action pour chaque action qui accorde des autorisations, choisissez le lien dans le récapitulatif du service. La table récapitulative de l'action inclut des détails concernant la ressource, notamment sa Région et son Compte. Vous pouvez également consulter les conditions qui s'appliquent à chaque ressource. Cela affiche les conditions s'appliquant à certaines ressources mais pas à d'autres.

Présentation des éléments d'un récapitulatif d'action

L'exemple ci-dessous illustre le récapitulatif de l'action `PutObject` (écriture) du récapitulatif de service Amazon S3 (consultez Récapitulatif du service (liste des actions)). Pour cette action, la politique définit plusieurs conditions sur une seule ressource.



La page de récapitulatif de l'action comprend les informations suivantes :

1. Choisissez JSON pour consulter des détails supplémentaires sur la politique, tels que les conditions multiples appliquées aux actions. (Si vous affichez le résumé d'action d'une politique en ligne qui est directement attachée à un utilisateur, les étapes diffèrent. Pour accéder au document de politique JSON dans ce cas, vous devez fermer la boîte de dialogue de résumé d'action et revenir au résumé de politique.)
2. Pour afficher le résumé d'une ressource spécifique, tapez des mots-clés dans la zone de Recherche afin de réduire la liste des ressources disponibles.
3. À côté de la flèche de retour des actions apparaissent le nom du service et de l'action au format `action name action in service` (dans ce cas, PutObject action dans S3). Le récapitulatif de l'action pour ce service inclut la liste des ressources autorisées définies dans la politique.
4. Resource (Ressource) : cette colonne présente les ressources que la politique définit pour le service choisi. Dans cet exemple, l'PutObjectaction est autorisée sur tous les chemins d'objets, mais uniquement sur la ressource du compartiment `developer_bucket` Amazon S3. En fonction des informations que le service fournit à IAM, vous verrez s'afficher un ARN, tel que `arn:aws:s3:::developer_bucket/*`, ou le type de ressource défini, tel que `BucketName = developer_bucket, ObjectPath = All`.
5. Region (Région) : cette colonne présente la région dans laquelle la ressource est définie. Il est possible de définir des ressources pour toutes les régions ou pour une seule région. Les ressources ne peuvent exister que dans une région spécifique.
   - Toutes les régions : les actions associées à la ressource s'appliquent à toutes les régions. Dans cet exemple, l'action appartient à un service mondial, Amazon S3. Les actions qui appartiennent aux services mondiaux s'appliquent à toutes les régions.

- Region text (Texte de région) : les actions associées à la ressource s'appliquent à une région. Par exemple, une politique peut spécifier la région `us-east-2` pour une ressource.

6. Account (Compte) : cette colonne indique si les services ou actions associés à la ressource s'appliquent à un compte spécifique. Les ressources peuvent exister dans tous les comptes ou dans un seul. Celles-ci ne peuvent pas exister dans un compte spécifique.
   - All accounts (Tous les comptes) : les actions associées à la ressource s'appliquent à tous les comptes. Dans cet exemple, l'action appartient à un service mondial, Amazon S3. Les actions qui appartiennent aux services mondiaux s'appliquent à tous les comptes.
   - Ce compte : les actions associées à la ressource s'appliquent uniquement au compte auquel vous êtes connecté.
   - Account number (Numéro du compte) : les actions associées à la ressource s'appliquent uniquement à un compte (un compte auquel vous n'êtes pas connecté). Par exemple, si une politique précise le compte `123456789012` pour une ressource, le numéro de compte apparaît dans le récapitulatif de la politique.
7. Request condition (Demander une condition) : cette colonne indique si les actions qui sont associées aux ressources font l'objet de conditions. Cet exemple inclut la condition `s3:x-amz-acl = public-read`. Pour en savoir plus sur ces conditions, choisissez JSON pour examiner le document de politique JSON.

Affichage des récapitulatifs des actions

Vous pouvez consulter un récapitulatif d'action pour chaque action répertorié dans le récapitulatif de la politique qui accorde les autorisations.

Affichage des récapitulatifs des actions à partir de la page Politiques

Vous pouvez afficher le récapitulatif d'action pour les politiques gérées.

Pour afficher le récapitulatif de l'action d'une politique gérée

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de la stratégie à afficher.
4. Sur la page Détails de la politique, affichez l'onglet Autorisations pour consulter le résumé de la politique.

5. Dans la liste des services du récapitulatif de la politique, choisissez le nom du service à afficher.
6. Dans la liste des actions du récapitulatif du service, choisissez le nom de l'action à afficher.

Affichage des récapitulatifs d'action d'une politique attachée à un utilisateur

Vous pouvez afficher le récapitulatif d'action pour toute politique attachée à un utilisateur.

Pour afficher le récapitulatif de l'action d'une politique attachée à un utilisateur

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Choisissez Utilisateurs dans le panneau de navigation.
3. Dans la liste des utilisateurs, choisissez le nom de l'utilisateur dont vous souhaitez afficher la politique.
4. Sur la page Récapitulatif de l'utilisateur, affichez l'onglet Autorisations pour afficher la liste des stratégies attachées à l'utilisateur directement ou via un groupe.
5. Dans la table des politiques de l'utilisateur, choisissez le nom de la politique que vous voulez afficher.

   Si vous vous trouvez sur la page Utilisateurs et vous choisissez d'afficher le résumé des services pour une politique attachée à cet utilisateur, vous êtes redirigé vers la page Politiques. Vous pouvez afficher les résumés des services uniquement sur la page Politiques.
6. Dans la liste des services du récapitulatif de la politique, choisissez le nom du service à afficher.

   > **Note**
   >
   > Si la politique que vous sélectionnez est une politique en ligne attachée directement à l'utilisateur, le tableau récapitulatif du service s'affiche. Si la politique est une politique en ligne attachée via un groupe, vous êtes dirigé vers le document de politique JSON de ce groupe. Si la stratégie est une stratégie gérée, vous êtes dirigé vers le récapitulatif du service de cette stratégie sur la page Stratégies.
7. Dans la liste des actions du récapitulatif du service, choisissez le nom de l'action à afficher.

Affichage des récapitulatifs d'action d'une politique attachée à un rôle

Vous pouvez afficher le récapitulatif d'action pour toute politique attachée à un rôle.

Pour afficher le récapitulatif d'action d'une politique attachée à un rôle

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Rôles.
3. Dans la liste des rôles, choisissez le nom du rôle dont vous souhaitez afficher la politique.
4. Sur la page Récapitulatif du rôle, affichez l'onglet Autorisations pour afficher la liste des stratégies attachées au rôle.
5. Dans la table des politiques pour le rôle, choisissez le nom de la politique que vous voulez afficher.

   Si vous vous trouvez sur la page Rôles et vous choisissez d'afficher le résumé des services pour une politique attachée à cet utilisateur, vous êtes redirigé vers la page Politiques. Vous pouvez afficher les résumés des services uniquement sur la page Politiques.
6. Dans la liste des services du récapitulatif de la politique, choisissez le nom du service à afficher.
7. Dans la liste des actions du récapitulatif du service, choisissez le nom de l'action à afficher.

Exemples de récapitulatifs de la politique

Les exemples suivants incluent des politiques JSON et leurs récapitulatifs de la politique associés, des récapitulatifs du service, ainsi que des récapitulatifs de l'action pour vous aider à comprendre les autorisations accordées par le biais d'une politique.

Politique 1 : DenyCustomerBucket

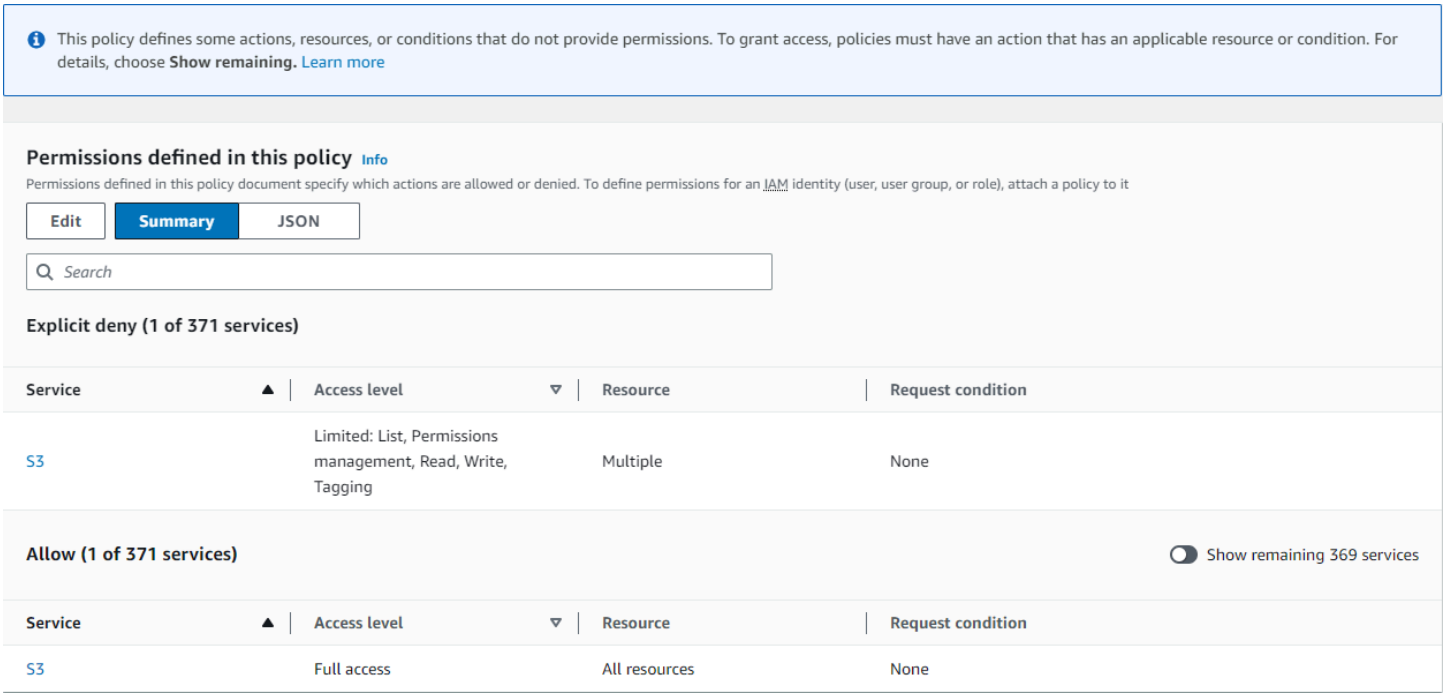
Cette politique illustre une autorisation et un refus pour le même service.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccess",
            "Effect": "Allow",
            "Action": ["s3:*"],
            "Resource": ["*"]
```

```
        },
        {
            "Sid": "DenyCustomerBucket",
            "Action": ["s3:*"],
            "Effect": "Deny",
            "Resource": ["arn:aws:s3:::customer", "arn:aws:s3:::customer/*" ]
        }
    ]
}
```

DenyCustomerBucketRésumé de la politique :



DenyCustomerBucket Résumé du service S3 (refus explicite) :

< Services Actions in S3 (82 of 130) Show remaining 48 actions

Read (35 of 53)

Action	Resource	Request condition
GetAccelerateConfiguration	BucketName| string like |customer	None
GetAnalyticsConfiguration	BucketName| string like |customer	None
GetBucketAcl	BucketName| string like |customer	None
GetBucketCORS	BucketName| string like |customer	None
GetBucketLocation	BucketName| string like |customer	None
GetBucketLogging	BucketName| string like |customer	None
GetBucketNotification	BucketName| string like |customer	None
GetBucketObjectLockConfiguration	BucketName| string like |customer	None
GetBucketOwnershipControls	BucketName| string like |customer	None
GetBucketPolicy	BucketName| string like |customer	None
GetBucketPolicyStatus	BucketName| string like |customer	None
GetBucketPublicAccessBlock	BucketName| string like |customer	None
GetBucketRequestPayment	BucketName| string like |customer	None
GetBucketTagging	BucketName| string like |customer	None
GetBucketVersioning	BucketName| string like |customer	None
GetBucketWebsite	BucketName| string like |customer	None

GetObject (Lire) Résumé des mesures :

< Actions GetObject action in S3

Resource	Region	Account	Request condition
BucketName| string like |customer, ObjectPath| string like |All	-	All accounts	None

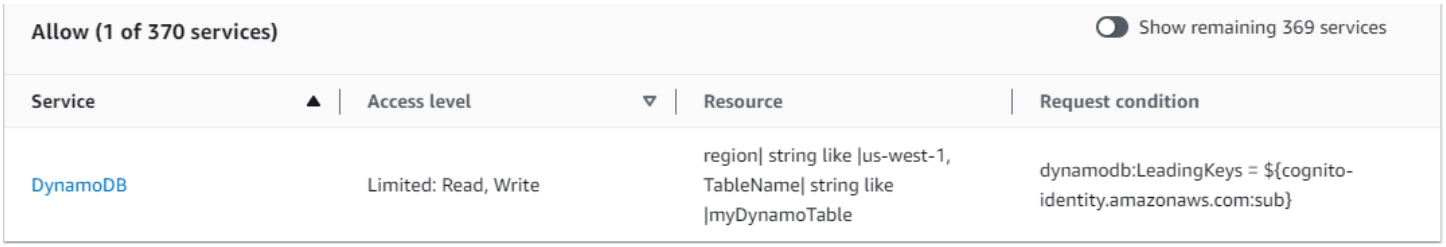
Politique 2 : DynamoDbRowCognitoID

Cette politique offre un accès de niveau ligne à Amazon DynamoDB en fonction de l'ID Amazon Cognito de l'utilisateur.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": [
                "arn:aws:dynamodb:us-west-1:123456789012:table/myDynamoTable"
            ],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": [
                        "${cognito-identity.amazonaws.com:sub}"
                    ]
                }
            }
        }
    ]
}
```

DynamoDbRowCognitoIDRésumé de la politique :



DynamoDbRowCognitoID Résumé du service DynamoDB (Autoriser) :

< Services Actions in DynamoDB (4 of 65) Show remaining 61 actions

Read (1 of 26)

Action	Resource	Request condition
GetItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

Write (3 of 33)

Action	Resource	Request condition
DeleteItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
PutItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
UpdateItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

GetItem Résumé des actions (Liste) :

< Actions GetItem action in DynamoDB

Resource	Region	Account	Request condition
region| string like |us-west-1, TableName| string like |myDynamoTable	us-west-1	123456789012	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

Politique 3 : MultipleResourceCondition

Cette politique comprend plusieurs ressources et conditions.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Apple_bucket/*"],
            "Condition": {"StringEquals": {"s3:x-amz-acl": ["public-read"]}}
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Orange_bucket/*"],
            "Condition": {"StringEquals": {
                "s3:x-amz-acl": ["custom"],
                "s3:x-amz-grant-full-control": ["1234"]
            }}
        }
    ]
}
```

MultipleResourceConditionRésumé de la politique :

Allow (1 of 370 services) Show remaining 369 services

Service	Access level	Resource	Request condition
S3	Limited: Permissions management, Write	Multiple	Multiple

MultipleResourceCondition Résumé du service S3 (Autoriser) :

< Services Actions in S3 (2 of 130) Show remaining 128 actions

Write (1 of 47)

Action	Resource	Request condition
PutObject	Multiple	Multiple

Permission Management (1 of 15)

Action	Resource	Request condition
PutObjectAcl	Multiple	Multiple

PutObject (Rédiger) le résumé de l'action :

< Actions PutObject action in S3

Resource	Region	Account	Request condition
Multiple	-	All accounts	Multiple

Politique 4 : EC2_troubleshoot

La politique suivante permet aux utilisateurs de récupérer une capture d'écran d'une instance Amazon EC2 en cours d'exécution, ce qui facilite la résolution des problèmes liés à EC2. Cette politique permet également de consulter des informations sur les éléments du compartiment du développeur Amazon S3.

JSON

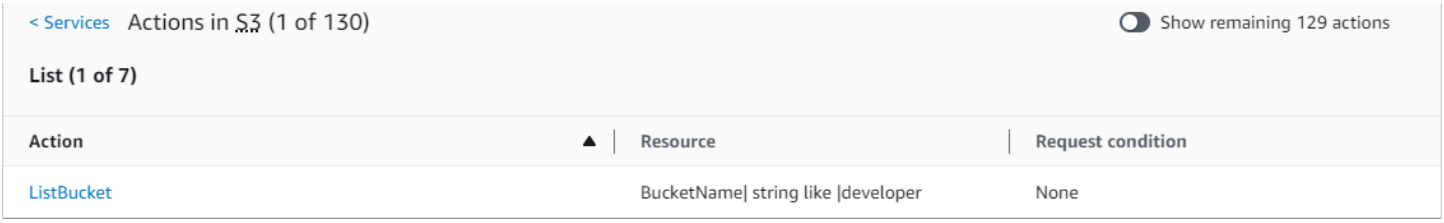
```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshot"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::developer"
            ]
        }
    ]
}
```

EC2_Troubleshoot Récapitulatif de la politique :

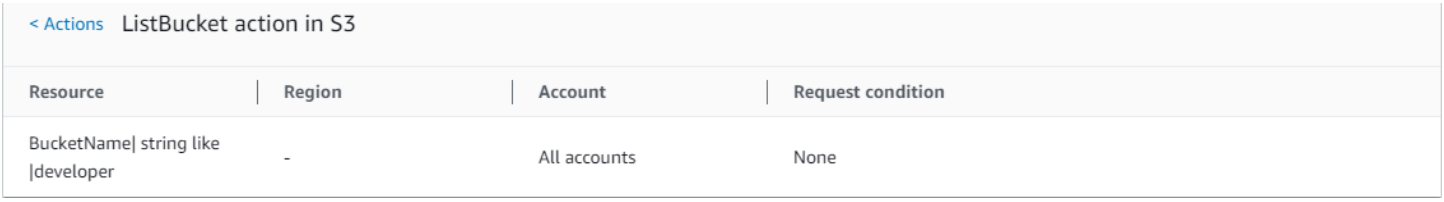
Allow (2 of 370 services) Show remaining 368 services

Service	Access level	Resource	Request condition
EC2	Limited: Read	All resources	None
S3	Limited: List	BucketName| string like |developer	None

EC2_Troubleshoot S3 (Allow) Récapitulatif du service :



ListBucket Résumé des actions (Liste) :



Politique 5 : CodeBuild _ CodeCommit _ CodeDeploy

Cette politique donne accès à des CodeDeploy ressources et à des ressources spécifiques CodeBuild. CodeCommit Ces ressources apparaissent uniquement avec le service correspondant car elles sont propres à chaque service. Si vous incluez une ressource qui ne correspond à aucun service de l'élément `Action`, elle apparaît dans tous les récapitulatifs de l'action.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1487980617000",
            "Effect": "Allow",
            "Action": [
                "codebuild:*",
                "codecommit:*",
                "codedeploy:*"
            ],
            "Resource": [
                "arn:aws:codebuild:us-east-2:123456789012:project/my-demo-
project",
                "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo",
                "arn:aws:codedeploy:us-
east-2:123456789012:application:WordPress_App",
                "arn:aws:codedeploy:us-east-2:123456789012:instance/AssetTag*"
```

```
            ]
        }
    ]
}
```

CodeBuild_ CodeCommit _ Résumé CodeDeploy de la politique :

Allow (3 of 370 services) Show remaining 367 services

Service	Access level	Resource	Request condition
CodeBuild	Full: Permissions management Limited: List, Read, Write	region| string like |us-east-2	None
CodeCommit	Full: Tagging Limited: List, Read, Write	ResourceSpecifier| string like |MyDemoRepo, region| string like |us-east-2	None
CodeDeploy	Full: Tagging Limited: List, Read, Write	Multiple	None

CodeBuild_ CodeCommit _ CodeDeploy CodeBuild (Autoriser) Résumé du service :

< Services Actions in CodeBuild (24 of 53) Show remaining 29 actions

Read (4 of 9)

Action	Resource	Request condition
BatchGetBuildBatches	region| string like |us-east-2	None
BatchGetBuilds	region| string like |us-east-2	None
BatchGetProjects	region| string like |us-east-2	None
GetResourcePolicy	region| string like |us-east-2	None

Write (16 of 28)

Action	Resource	Request condition
BatchDeleteBuilds	region| string like |us-east-2	None
CreateProject	region| string like |us-east-2	None
CreateWebhook	region| string like |us-east-2	None
DeleteBuildBatch	region| string like |us-east-2	None
DeleteProject	region| string like |us-east-2	None
DeleteWebhook	region| string like |us-east-2	None
InvalidateProjectCache	region| string like |us-east-2	None
RetryBuild	region| string like |us-east-2	None
RetryBuildBatch	region| string like |us-east-2	None
StartBuild	region| string like |us-east-2	None
StartBuildBatch	region| string like |us-east-2	None
StopBuild	region| string like |us-east-2	None
StopBuildBatch	region| string like |us-east-2	None
UpdateProject	region| string like |us-east-2	None
UpdateProjectVisibility	region| string like |us-east-2	None
UpdateWebhook	region| string like |us-east-2	None

List (2 of 14)

CodeBuild_ CodeCommit _ CodeDeploy StartBuild (Écrire) le résumé de l'action :

< Actions StartBuild action in CodeBuild

Resource	Region	Account	Request condition
region| string like |us-east-2	us-east-2	123456789012	None

Autorisations requises pour accéder aux autres ressources IAM

Les ressources sont des objets au sein d'un service. Les ressources IAM incluent des groupes, des utilisateurs, des rôles et des politiques. Si vous êtes connecté avec des Utilisateur racine d'un compte AWS informations d'identification, vous n'êtes soumis à aucune restriction quant à l'administration des informations d'identification IAM ou des ressources IAM. Cependant, les utilisateurs IAM doivent explicitement accorder des autorisations pour gérer les informations d'identification ou des ressources IAM. Pour ce faire, vous pouvez attacher une politique basée sur les identités à l'utilisateur.

> **Note**
>
> Tout au long de la AWS documentation, lorsque nous faisons référence à une politique IAM sans mentionner aucune catégorie spécifique, nous entendons une politique gérée par le client et basée sur l'identité. Pour de plus amples informations sur les catégories de politique, veuillez consulter the section called "Politiques et autorisations".

Autorisations pour la gestion des identités IAM

Les autorisations qui sont requises pour administrer des groupes, des utilisateurs, des rôles et des informations d'identification IAM correspondent aux actions d'API de la tâche. Par exemple, pour créer des utilisateurs IAM, vous devez disposer de l'autorisation `iam:CreateUser` qui comporte la commande API correspondante : `CreateUser`. Pour autoriser un utilisateur IAM à créer d'autres utilisateurs IAM, vous pouvez attacher une politique IAM comme celle qui suit à cet utilisateur :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

Dans une politique, la valeur de l'élément `Resource` dépend de l'action et des ressources que cette action peut affecter. Dans l'exemple précédent, la politique autorise un utilisateur à créer un autre utilisateur (* est un caractère générique qui correspond à toutes les chaînes). Par opposition, une politique qui autorise des utilisateurs à modifier uniquement leurs propres clés d'accès (actions API `CreateAccessKey` et `UpdateAccessKey`) comporte généralement un élément `Resource`. Dans ce cas, l'ARN inclut une variable (`${aws:username}`) qui se résout par le nom de l'utilisateur actuel, comme dans l'exemple suivant :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListUsersForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "arn:aws:iam::*:*"
        },
        {
            "Sid": "ViewAndUpdateAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:UpdateAccessKey",
                "iam:CreateAccessKey",
                "iam:ListAccessKeys"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Dans l'exemple précédent, `${aws:username}` est une variable qui renvoie au nom utilisateur de l'utilisateur actuel. Pour de plus amples informations sur les variables de politique, veuillez consulter Éléments des politiques IAM : variables et balises.

L'utilisation d'un caractère générique (*) dans le nom d'action facilite souvent l'octroi d'autorisations pour toutes les actions associées à une tâche spécifique. Par exemple, pour autoriser des utilisateurs à exécuter n'importe quelle action IAM, vous pouvez utiliser `iam:*` pour l'action. Pour

autoriser les utilisateurs à effectuer n'importe quelle action associée uniquement aux clés d'accès, vous pouvez utiliser `iam:*AccessKey*` dans l'élément `Action` d'une instruction de politique. Cela permet d'accorder à l'utilisateur l'autorisation d'effectuer les actions `CreateAccessKey`, `DeleteAccessKey`, `GetAccessKeyLastUsed`, `ListAccessKeys`, et `UpdateAccessKey`. (Si, dans le futur, une action dont le nom est « AccessKey » est ajoutée à IAM, l'utilisation de `iam:*AccessKey*` for pour l'`Action`élément autorisera également l'utilisateur à effectuer cette nouvelle action.) L'exemple suivant montre une politique qui permet aux utilisateurs d'effectuer toutes les actions relatives à leurs propres clés d'accès (à *`account-id`* remplacer par votre Compte AWS identifiant) :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::111122223333:user/${aws:username}"
    }
}
```

Certaines tâches, comme la suppression d'un groupe, impliquent plusieurs actions : vous devez d'abord supprimer les utilisateurs du groupe, puis détacher ou supprimer les stratégie du groupe et enfin supprimer réellement le groupe. Si vous souhaitez qu'un utilisateur supprime un groupe, vous devez vous assurer d'accorder à celui-ci les autorisations nécessaires pour exécuter toutes les actions associées.

Autorisations pour travailler dans Console de gestion AWS

Les exemples précédents illustrent des politiques autorisant un utilisateur à exécuter les actions avec l'interface AWS CLI ou les kits SDK AWS.

À mesure que les utilisateurs se servent de la console, celle-ci envoie des demandes à IAM pour répertorier les groupes, les utilisateurs, les rôles et les politiques, et pour obtenir les politiques associées à un groupe, un utilisateur ou un rôle. La console émet également des demandes pour obtenir Compte AWS des informations et des informations sur le principal. Le principal est l'utilisateur qui effectue les demandes sur la console.

En général, pour effectuer une action, vous devez posséder uniquement l'action correspondante dans une politique. Pour pouvoir créer un utilisateur, vous devez disposer d'une autorisation pour appeler l'action `CreateUser`. Souvent, lorsque vous utilisez la console pour effectuer une action, vous devez avoir les autorisations pour afficher, répertorier, obtenir, ou encore afficher les ressources sur la console. Ceci est nécessaire pour vous permettre de naviguer au sein de la console pour effectuer l'action spécifiée. Par exemple, si l'utilisateur Jorge souhaite utiliser la console pour modifier ses propres clés d'accès, il accède à la console IAM et choisit Users. Cette action indique à la console d'effectuer une demande `ListUsers`. Si Jorge n'a pas l'autorisation d'exécuter l'action `iam:ListUsers`, la console se voit refuser l'accès lorsqu'elle essaie de répertorier les utilisateurs. Par conséquent, Jorge ne peut accéder à son propre nom et à ses propres clés d'accès, même s'il a l'autorisation d'exécuter les actions `CreateAccessKey` et `UpdateAccessKey`.

Si vous souhaitez autoriser les utilisateurs à administrer des groupes, des utilisateurs, des rôles, des politiques et des informations d'identification avec le Console de gestion AWS, vous devez inclure des autorisations pour les actions effectuées par la console. Pour obtenir des exemples de politiques que vous pouvez utiliser pour accorder ces autorisations à un utilisateur, consultez Exemples de politiques de gestion de ressources IAM.

Accordez des autorisations à AWS comptes

Vous pouvez accorder directement à des utilisateurs IAM de votre propre compte un accès à vos ressources. Si des utilisateurs d'un autre compte ont besoin d'accéder à vos ressources, vous pouvez créer un rôle IAM qui est une entité incluant des autorisations, mais qui n'est pas associée à un utilisateur spécifique. Les utilisateurs d'autres comptes peuvent alors utiliser le rôle et accéder aux ressources selon les autorisations que vous avez attribuées à ce rôle. Pour plus d'informations, veuillez consulter Accès d'un utilisateur IAM à un autre Compte AWS que tu possèdes.

> **Note**
>
> Certains services prennent en charge les politiques basées sur les ressources, comme décrit dansIdentity-based politiques et politiques basées sur les ressources (Amazon S3, Amazon SNS et Amazon SQS, par ex.). Pour ces services, une alternative à l'utilisation de rôles consiste à attacher une politique à la ressource (compartiment, rubrique ou file d'attente) que vous voulez partager. La politique basée sur les ressources peut spécifier le AWS compte autorisé à accéder à la ressource.

Autorisations pour qu'un service accède à un autre service

De nombreux AWS services accèdent à d'autres AWS services. Par exemple, plusieurs services AWS , notamment Amazon EMR, Elastic Load Balancing et Amazon EC2 Auto Scaling, gèrent des instances Amazon EC2. D'autres AWS services utilisent les compartiments Amazon S3, les rubriques Amazon SNS, les files d'attente Amazon SQS, etc.

Le scénario pour gérer des autorisations dans ces cas varie selon le service. Voici des exemples de la façon dont les autorisations sont gérées pour différents services :

- Dans Amazon EC2 Auto Scaling, les utilisateurs doivent être autorisés à utiliser Auto Scaling, mais n'ont pas besoin de recevoir explicitement l'autorisation de gérer des instances Amazon EC2.
- Dans AWS Data Pipeline, un rôle IAM détermine ce que peut faire un pipeline ; les utilisateurs ont besoin d'une autorisation pour assumer ce rôle. (Pour obtenir des détails, veuillez consulter Granting Permissions to Pipelines with IAM (Octroi d'autorisations à des pipelines avec IAM)dans le Manuel du développeur AWS Data Pipeline .)

Pour plus de détails sur la manière de configurer correctement les autorisations afin qu'un AWS service puisse accomplir les tâches que vous souhaitez, consultez la documentation du service que vous appelez. Pour savoir comment créer un rôle pour un service, consultez Créez un rôle pour déléguer des autorisations à un AWS web.

Configuration d'un service avec un rôle IAM pour l'exécution d'une tâche en votre nom

Lorsque vous souhaitez configurer un AWS service pour qu'il fonctionne en votre nom, vous fournissez généralement l'ARN d'un rôle IAM qui définit ce que le service est autorisé à faire. AWS vérifie que vous êtes autorisé à transmettre un rôle à un service. Pour de plus amples informations, veuillez consulter Accorder à un utilisateur l'autorisation de transmettre un rôle à un AWS web.

Actions requises

Les actions sont les choses que vous pouvez faire sur une ressource, comme l'affichage, la création, l'édition et la suppression de cette ressource. Les actions sont définies par chaque AWS service.

Pour autoriser une personne à effectuer une action, vous devez inclure les actions nécessaires dans une politique qui s'applique à l'identité appelante ou à la ressource affectée. En général, pour fournir l'autorisation requise pour effectuer une action, vous devez inclure cette action dans votre politique. Par exemple, pour créer un utilisateur, vous devez ajouter l' CreateUser action à votre politique.

Dans certains cas, une action peut exiger l'inclusion d'actions connexes supplémentaires dans votre politique. Par exemple, pour fournir à une personne l'autorisation de créer un annuaire dans AWS Directory Service à l'aide de l'opération `ds:CreateDirectory`, vous devez inclure les actions suivantes dans sa politique :

- `ds:CreateDirectory`
- `ec2:DescribeSubnets`
- `ec2:DescribeVpcs`
- `ec2:CreateSecurityGroup`
- `ec2:CreateNetworkInterface`
- `ec2:DescribeNetworkInterfaces`
- `ec2:AuthorizeSecurityGroupIngress`
- `ec2:AuthorizeSecurityGroupEgress`

Lorsque vous créez ou éditez une politique à l'aide de l'éditeur visuel, vous recevez des avertissements et des instructions pour vous aider à choisir toutes les actions requises pour votre politique.

Pour plus d'informations sur les autorisations requises pour créer un répertoire dans AWS Directory Service, voir Exemple 2 : Autoriser un utilisateur à créer un répertoire.

Exemples de politiques de gestion de ressources IAM

Voici des exemples de politiques IAM qui autorisent les utilisateurs à exécuter des tâches associées à la gestion des utilisateurs, des groupes et des informations d'identification IAM. Il s'agit notamment de politiques qui permettent aux utilisateurs de gérer leurs propres mots de passe, clés d'accès et dispositifs d'authentification multi-facteur (MFA).

Pour des exemples de politiques qui permettent aux utilisateurs d'effectuer des tâches avec d'autres AWS services, tels qu'Amazon S3, Amazon EC2 et DynamoDB, consultez. Exemples de politiques basées sur l'identité IAM

Rubriques

- Autoriser un utilisateur à répertorier les groupes, les utilisateurs et les politiques d'un compte, ainsi que d'autres informations à des fins d'élaboration de rapports
- Autoriser un utilisateur à gérer l'adhésion à un groupe

- Autoriser un utilisateur à gérer les utilisateurs IAM
- Autoriser les utilisateurs à définir la politique de mot de passe du compte
- Autoriser les utilisateurs à générer et extraire des rapports d'informations d'identification IAM
- Autoriser toutes les actions IAM (Accès Admin)

Autoriser un utilisateur à répertorier les groupes, les utilisateurs et les politiques d'un compte, ainsi que d'autres informations à des fins d'élaboration de rapports

La politique suivante permet à l'utilisateur d'appeler toute action IAM qui commence par la chaîne `Get` ou `List` et générer des rapports. Pour afficher l'exemple de politique, consultez IAM : autorise l'accès en lecture seule à la console IAM.

Autoriser un utilisateur à gérer l'adhésion à un groupe

La politique suivante permet à l'utilisateur de mettre à jour l'appartenance au groupe appelé MarketingGroup. Pour afficher l'exemple de politique, consultez IAM : autorise la gestion des membres d'un groupe par programmation et dans la console.

Autoriser un utilisateur à gérer les utilisateurs IAM

La politique suivante permet à un utilisateur d'exécuter toutes les tâches associées à la gestion des utilisateurs IAM, mais pas d'effectuer d'actions sur d'autres entités, par exemple la création de groupes ou de politiques. Les actions autorisées sont notamment :

- Création de l'utilisateur (action `CreateUser`).
- Suppression de l'utilisateur. Cette tâche requiert des autorisations pour effectuer toutes les actions suivantes : `DeleteSigningCertificate`, `DeleteLoginProfile`, `RemoveUserFromGroup` et `DeleteUser`.
- Affichage de la liste des utilisateurs du compte et des groupes (les actions `GetUser`, `ListUsers` et `ListGroupsForUser`).
- Affichage de la liste et suppression des politiques pour l'utilisateur (les actions `ListUserPolicies`, `ListAttachedUserPolicies`, `DetachUserPolicy`, `DeleteUserPolicy`)
- Changement de nom ou modification du chemin d'accès pour l'utilisateur (action `UpdateUser`). L'élément `Resource` doit inclure un ARN qui inclut à la fois le chemin d'accès source et le chemin

d'accès cible. Pour plus d'informations sur les chemins d'accès, consultez Noms conviviaux et chemins.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowUsersToPerformUserActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListPolicies",
                "iam:GetPolicy",
                "iam:UpdateUser",
                "iam:AttachUserPolicy",
                "iam:ListEntitiesForPolicy",
                "iam:DeleteUserPolicy",
                "iam:DeleteUser",
                "iam:ListUserPolicies",
                "iam:CreateUser",
                "iam:RemoveUserFromGroup",
                "iam:AddUserToGroup",
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:PutUserPolicy",
                "iam:ListAttachedUserPolicies",
                "iam:ListUsers",
                "iam:GetUser",
                "iam:DetachUserPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUsersToSeeStatsOnIAMConsoleDashboard",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccount*",
                "iam:ListAccount*"
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

Plusieurs autorisations incluses dans la politique précédente permettent à l'utilisateur d'exécuter des tâches dans Console de gestion AWS. Les utilisateurs qui exécutent des tâches liées à l'utilisateur à partir de la AWS CLI, des kits SDK AWS ou de l'API Query HTTP IAM uniquement peuvent ne pas avoir besoin de certaines autorisations. Par exemple, si les utilisateurs connaissent déjà l'ARN des politiques à détacher d'un utilisateur, ils n'ont pas besoin d'autorisation `iam:ListAttachedUserPolicies`. La liste exacte des autorisations requises par un utilisateur varie en fonction des tâches qu'il doit effectuer lorsqu'il gère d'autres utilisateurs.

Les autorisations suivantes de la politique permettent l'accès aux tâches utilisateur via Console de gestion AWS:

- `iam:GetAccount*`
- `iam:ListAccount*`

Autoriser les utilisateurs à définir la politique de mot de passe du compte

Vous pouvez octroyer à certains utilisateurs des autorisations pour obtenir et mettre à jour la politique de mot de passe de votre Compte AWS. Pour afficher l'exemple de politique, consultez IAM : permet de définir les exigences de mot de passe du compte par programmation et dans la console.

Autoriser les utilisateurs à générer et extraire des rapports d'informations d'identification IAM

Vous pouvez autoriser les utilisateurs à générer et à télécharger un rapport répertoriant tous les utilisateurs de votre compte Compte AWS. Le rapport indique également l'état de plusieurs informations d'identification utilisateur, y compris les mots de passe, les clés d'accès, les dispositifs MFA et les certificats de signature. Pour de plus amples informations sur les rapports d'informations d'identification, veuillez consulter Générez des rapports d'identification pour votre Compte AWS. Pour afficher l'exemple de politique, consultez IAM : générer et extraire des rapports sur les informations d'identification IAM.

Autoriser toutes les actions IAM (Accès Admin)

Il est possible d'accorder à certains utilisateurs des autorisations administratives leur permettant d'effectuer toutes les actions dans IAM, y compris la gestion des mots de passe et des clés d'accès,

des dispositifs MFA et des certificats utilisateur. L'exemple de politique suivant accorde ces autorisations.

> **Warning**
>
> Lorsque vous accordez à un utilisateur un accès complet à IAM, il n'y a aucune limite aux autorisations que l'utilisateur peut accorder à him/herself ou à d'autres personnes. Ainsi, l'utilisateur peut créer de nouvelles entités IAM (utilisateurs ou rôles) et leur accorder un accès total à toutes les ressources de votre Compte AWS. Lorsque vous octroyez à un utilisateur un accès complet à IAM, vous lui donnez en réalité un accès total à toutes les ressources de votre Compte AWS. Cela inclut la possibilité de supprimer toutes les ressources. Vous devez octroyer ces autorisations aux administrateurs approuvés uniquement, et vous devez également mettre en œuvre l'authentification multi-facteur (MFA) pour ces administrateurs.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*",
    "Resource": "*"
  }
}
```

Délégation temporaire de l'IAM

Présentation de

La délégation temporaire accélère l'intégration et simplifie la gestion des produits d'Amazon et de AWS ses partenaires qui s'intègrent à vos AWS comptes. Au lieu de configurer manuellement plusieurs AWS services, vous pouvez déléguer des autorisations temporaires et limitées qui permettent au fournisseur du produit d'effectuer les tâches de configuration en votre nom en quelques minutes grâce à des flux de déploiement automatisés. Vous conservez le contrôle administratif en ce qui concerne les exigences d'approbation et les limites des autorisations,

tandis que les autorisations des fournisseurs de produits expirent automatiquement après la durée approuvée, sans qu'aucun nettoyage manuel ne soit nécessaire. Si le produit nécessite un accès permanent pour les opérations en cours, le fournisseur peut utiliser la délégation temporaire pour créer un rôle IAM avec une limite d'autorisation qui définit les autorisations maximales du rôle. Toutes les activités des fournisseurs de produits sont suivies à des AWS CloudTrail fins de surveillance de la conformité et de la sécurité.

Pour traiter les demandes de délégation temporaire, vous pouvez lancer une demande et examiner les demandes. Pour plus d'informations sur la surveillance, consultez la section Notifications et CloudTrail journalisation.

> **Note**
>
> Les demandes de délégation temporaires ne peuvent être créées que par des produits Amazon et AWS des partenaires qualifiés ayant terminé le processus d'intégration des fonctionnalités. Les clients vérifient et approuvent ces demandes mais ne peuvent pas les créer directement. Si vous êtes un AWS partenaire qui souhaite intégrer la délégation temporaire IAM dans votre produit, consultez le Guide d'intégration des partenaires pour obtenir des instructions d'intégration et d'intégration.

Comment fonctionne la délégation temporaire

La délégation temporaire permet à Amazon et à AWS ses partenaires de demander un accès temporaire et limité à votre compte. Après votre approbation, ils peuvent utiliser des autorisations déléguées pour prendre des mesures en votre nom. Les demandes de délégation définissent des autorisations spécifiques pour les AWS services et les actions dont le fournisseur du produit a besoin pour déployer ou configurer les ressources de votre AWS compte. Ces autorisations ne sont disponibles que pour une durée limitée et expirent automatiquement après la durée spécifiée dans la demande.

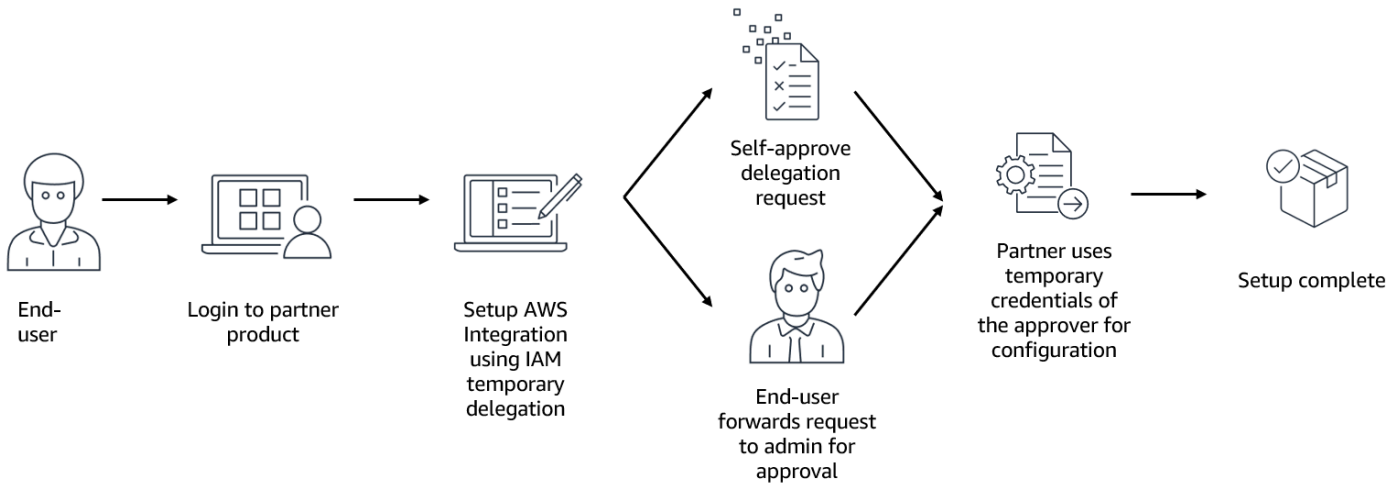
> **Note**
>
> La durée maximale de l'accès délégué est de 12 heures. Toutefois, les utilisateurs root ne peuvent approuver que les demandes de délégation d'une durée inférieure ou égale à 4 heures. Si une demande nécessite plus de 4 heures, vous devez utiliser une identité non root

pour approuver la demande. Pour plus de détails, consultez la section Fonctionnalité bêta de simulation d'autorisations.

Pour les tâches en cours, telles que la lecture depuis un compartiment Amazon S3, les demandes de délégation peuvent inclure la création d'un rôle IAM qui permet un accès continu aux ressources et aux actions après l'expiration de l'accès temporaire. Les fournisseurs de produits doivent attribuer une limite d'autorisation à tout rôle IAM créé par délégation temporaire. Les limites d'autorisation limitent les autorisations maximales d'un rôle mais n'accordent pas d'autorisations à elles seules. Vous pouvez consulter la limite d'autorisation dans le cadre de la demande avant de l'approuver. Pour plus de détails, consultez la section Limites des autorisations.

Le processus fonctionne comme suit :

1. Vous vous connectez à un produit Amazon ou AWS partenaire pour l'intégrer à votre AWS environnement.
2. Le fournisseur du produit lance une demande de délégation en votre nom et vous redirige vers la console de AWS gestion.
3. Vous passez en revue les autorisations demandées et déterminez si vous souhaitez approuver, refuser ou transmettre la demande à votre administrateur.
4. Une fois que vous ou votre administrateur avez approuvé la demande, le fournisseur du produit peut obtenir les informations d'identification temporaires de l'approbateur pour effectuer les tâches requises.
5. L'accès des fournisseurs de produits expire automatiquement après la période spécifiée. Cependant, tout rôle IAM créé par la demande de délégation temporaire persiste au-delà de cette période, ce qui permet au fournisseur du produit de continuer à accéder aux ressources et aux actions pour les tâches de gestion en cours.



> **Note**
>
> Vous ne pouvez déléguer des autorisations à un fournisseur de produits que si vous êtes autorisé à accéder aux services et aux actions inclus dans la demande de délégation temporaire. Si vous n'avez pas accès aux services et actions demandés, le fournisseur du produit ne reçoit pas ces autorisations lorsque vous approuvez la demande.

Si la vérification des autorisations montre qu'elle est susceptible de réussir, vous pouvez approuver la demande de délégation temporaire et poursuivre le flux de travail.

Si le contrôle des autorisations indique que vous ne disposez peut-être pas des autorisations suffisantes, transmettez la demande à votre administrateur pour approbation. Nous vous recommandons d'informer votre administrateur de cette demande en utilisant la méthode de votre choix, par exemple un e-mail ou un ticket.

Une fois que votre administrateur a approuvé la demande, ce qui se passe ensuite dépend de la configuration du fournisseur du produit :

- Si le fournisseur du produit a demandé un accès immédiat, il reçoit automatiquement des autorisations temporaires et la durée d'accès commence.
- Si le fournisseur du produit a demandé la libération par le propriétaire (destinataire initial), vous devez revenir à la demande pour partager explicitement l'accès temporaire au compte avant le début de la durée d'accès. Les fournisseurs de produits utilisent généralement cette option

lorsqu'ils ont besoin d'informations supplémentaires de votre part, telles que la sélection des ressources ou les détails de configuration, pour effectuer la tâche requise.

Lancer une demande de délégation temporaire

Vous pouvez lancer une demande de délégation temporaire uniquement à partir de produits Amazon ou AWS partenaires pris en charge. Au cours d'un flux de travail qui prend en charge la délégation temporaire, vous serez invité à accorder au fournisseur du produit des autorisations temporaires et limitées pour configurer les AWS ressources requises sur votre compte. Cette approche automatisée fournit une expérience plus rationalisée en vous évitant de configurer ces ressources manuellement.

Vous pouvez consulter les détails des demandes de délégation, tels que les rôles, les politiques et les AWS services IAM spécifiques dont le fournisseur de produits a besoin avant d'accorder l'accès. Vous pouvez soit approuver la demande vous-même si vous disposez des autorisations suffisantes, soit transmettre la demande à l'administrateur de votre compte pour approbation. Tous les accès des fournisseurs de produits sont limités dans le temps et peuvent être surveillés et révoqués si nécessaire.

Pour lancer une demande de délégation temporaire

1. Accédez à la console d'un produit pris en charge par Amazon ou AWS ses partenaires qui nécessite une intégration à votre AWS compte.
2. Sélectionnez Déployer avec délégation temporaire IAM. Notez que le nom de l'option peut varier selon les produits pris en charge. Reportez-vous à la documentation du fournisseur du produit pour plus de détails.

   > **Note**
   >
   > Si vous n'êtes pas encore connecté à la console AWS de gestion, une nouvelle fenêtre s'ouvre sur la page de AWS connexion. Nous vous recommandons de vous connecter à votre AWS compte avant de lancer la demande de délégation temporaire depuis la console du produit. Pour plus d'informations sur la procédure de connexion en fonction de votre type d'utilisateur et des AWS ressources auxquelles vous souhaitez accéder, consultez le Guide de AWS Sign-In l'utilisateur.

3. Vérifiez les détails de la demande pour confirmer le nom du produit et le AWS compte du fournisseur du produit. Vous pouvez également vérifier l' AWS identité que le fournisseur du produit utilisera pour effectuer des actions en votre nom.

4. Consultez les détails d'accès pour les autorisations qui seront temporairement déléguées en approuvant cette demande.

   - La section Résumé des autorisations fournit une vue d'ensemble de haut niveau générée par l'IA qui peut vous aider à comprendre quelles catégories de AWS services sont accessibles et quels types d'actions peuvent être effectués dans chaque service.
   - Choisissez Afficher le JSON pour consulter les autorisations spécifiques dont le fournisseur du produit a besoin pour déployer sur votre AWS compte, notamment l'étendue de l'accès et les limites de ressources.
   - Si le fournisseur du produit crée un rôle IAM dans le cadre d'une demande de délégation temporaire, une limite d'autorisation doit être attachée au rôle. Ces rôles IAM disposent d'autorisations qui continuent à autoriser l'accès aux ressources et aux actions après l'expiration de la durée d'accès demandée. Choisissez Afficher les détails pour consulter la limite des autorisations, qui définit les autorisations maximales que le rôle peut avoir. Le fournisseur du produit appliquera des politiques supplémentaires au rôle lors de sa création afin de définir ses autorisations réelles. Ces politiques peuvent sembler plus ciblées ou plus larges que les limites, selon la façon dont le fournisseur de produits les définit. Toutefois, la limite d'autorisation garantit que les autorisations effectives du rôle ne dépasseront jamais ce que vous voyez lors de l'approbation de la demande, quelles que soient les politiques associées au rôle. Pour plus d'informations, consultez la section Limites des autorisations.

5. Passez en revue les résultats de la simulation des autorisations. La fonction de simulation des autorisations évalue automatiquement les autorisations de votre identité par rapport à celles incluses dans la demande. Sur la base de cette analyse, une recommandation s'affiche indiquant s'il faut approuver la demande avec votre identité actuelle ou la transmettre à un administrateur. Pour plus de détails, consultez la section Fonctionnalité bêta de simulation d'autorisations.

6. Dans la boîte de dialogue, sélectionnez la manière dont vous souhaitez procéder.

   - Sélectionnez Autoriser l'accès lorsque votre identité dispose des autorisations suffisantes pour permettre au fournisseur du produit d'effectuer les procédures d'intégration en votre nom. Lorsque vous sélectionnez cette option, la durée d'accès du fournisseur du produit commence une fois que vous avez fourni l'accès.
   - Sélectionnez Demander une approbation si votre identité ne dispose pas des autorisations suffisantes pour permettre au fournisseur du produit d'effectuer les procédures d'intégration en votre nom. Choisissez ensuite Créer une demande d'approbation. Lorsque vous sélectionnez cette option, un lien de demande de délégation temporaire est créé et vous pouvez le partager avec l'administrateur de votre compte. Votre administrateur peut accéder à la console de AWS

gestion ou utiliser le lien d'accès pour examiner les demandes de délégation temporaire afin d'approuver la demande et de partager l'accès temporaire avec le demandeur.

> **Note**
>
> L'octroi de l'accès au fournisseur de produits nécessite deux actions : accepter la demande de délégation (`AcceptDelegationRequest`) et libérer le jeton d'échange (`SendDelegationToken`). La console AWS de gestion exécute les deux étapes automatiquement lorsque vous approuvez une demande. Si vous utilisez l'API AWS CLI or, vous devez exécuter les deux étapes séparément.

Capacité de simulation d'autorisations -bêta

Lorsque vous recevez une demande de délégation temporaire, vous pouvez l'approuver vous-même ou la transmettre à l'administrateur de votre compte pour approbation. Vous ne pouvez déléguer des autorisations à un fournisseur de produits que si vous êtes autorisé à accéder aux services et aux actions inclus dans la demande de délégation temporaire. Si vous n'avez pas accès aux services et actions demandés, le fournisseur du produit ne recevra pas ces autorisations, même si elles sont incluses dans la demande.

Par exemple, une demande de délégation temporaire nécessite la possibilité de créer un compartiment Amazon S3, de démarrer et d'arrêter des instances dans Amazon EC2 et d'assumer un rôle IAM. L'identité qui approuve la demande peut démarrer et arrêter des instances dans Amazon EC2 et assumer un rôle IAM, mais n'est pas autorisée à créer un compartiment Amazon S3. Lorsque cette identité approuve la demande, le fournisseur du produit n'est pas en mesure de créer un compartiment Amazon S3 même si ces autorisations étaient incluses dans la demande de délégation temporaire.

Comme vous ne pouvez déléguer que les autorisations que vous possédez déjà, il est essentiel d'évaluer si vous disposez des autorisations demandées avant de les approuver. La fonctionnalité bêta de simulation des autorisations facilite cette évaluation en comparant vos autorisations à celles incluses dans la demande. L'évaluation indique si vous pouvez approuver la demande avec votre identité actuelle ou si vous devez la transmettre à un administrateur. Si l'analyse ne permet pas de confirmer que vous disposez des autorisations suffisantes, transférez la demande à un administrateur pour examen. Cette évaluation est basée sur une analyse des autorisations simulée et peut différer

de votre AWS environnement réel. Vérifiez donc attentivement les autorisations demandées avant de poursuivre.

Étapes suivantes

Après avoir lancé une demande de délégation temporaire, vous pouvez gérer et surveiller la demande tout au long de son cycle de vie. Les procédures suivantes vous permettent de suivre, d'approuver et de contrôler les accès temporaires :

- Consultez les demandes de délégation temporaire : surveillez l'état de vos demandes d'accès et consultez des informations détaillées sur les demandes d'approbation ou de refus des demandes de délégation temporaire.
- Révoquer l'accès à la délégation temporaire — Mettez immédiatement fin aux sessions de délégation temporaires actives avant leur expiration naturelle.

Examiner les demandes de délégation temporaire

Après avoir lancé une demande de délégation temporaire, vous pouvez surveiller, approuver et rejeter les demandes dans la console IAM. La page Demandes de délégation temporaire fournit une vue centralisée de toutes les demandes, y compris celles en attente d'approbation, terminées ou rejetées. En tant qu'administrateur, vous pouvez examiner ces demandes pour accorder aux fournisseurs de produits l'accès aux AWS ressources ou les rejeter en fonction des politiques de sécurité, des exigences commerciales ou des normes de conformité de votre organisation. Cette visibilité vous permet de suivre le cycle de vie de l'accès des fournisseurs de produits et de contrôler les autorisations temporaires.

> **Note**
>
> Vous devez disposer de l'AcceptDelegationRequest autorisation iam : pour approuver les demandes de délégation temporaire.

Pour approuver une demande de délégation temporaire

1. Connectez-vous à la console de AWS gestion et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le volet de navigation de gauche, sélectionnez Demandes de délégation temporaires.

3. La page principale affiche la liste de vos demandes de délégation temporaire avec les informations suivantes :

   - ID de demande — Identifiant unique de la demande
   - État — État actuel (En attente, approuvé, rejeté, partagé, expiré)
   - Demandeur : fournisseur du produit associé à la demande
   - Initié par : le responsable IAM du compte qui a initié la demande pour le fournisseur du produit
   - Demande créée — Quand la demande a été soumise
   - Expiration de la demande : date à laquelle la demande a expiré ou expirera

4. (Facultatif) Utilisez les options de filtrage pour afficher les demandes par statut :

   - Toutes les demandes — Consultez toutes vos demandes, quel que soit leur statut
   - En attente — Afficher les demandes en attente d'approbation de l'administrateur
   - Approuvé — Afficher les demandes approuvées
   - Partagé — Afficher les demandes pour lesquelles l'accès a été partagé
   - Rejeté — Afficher les demandes rejetées avec les raisons du rejet

5. Pour afficher des informations détaillées sur une demande spécifique ou pour examiner une demande d'approbation en attente, choisissez l'ID de la demande.

6. Consultez les informations détaillées de la demande :

   - Informations sur le fournisseur du produit
   - Motif et justification de la demande
   - Durée demandée
   - AWS Autorisations demandées

7. Si vous êtes administrateur et que vous examinez une demande en attente, choisissez l'une des options suivantes :

   - Pour approuver la demande, choisissez Approuver. Dans la boîte de dialogue d'approbation, vous pouvez consulter les résultats de la simulation des autorisations. Pour plus d'informations, consultez la section Fonctionnalité bêta de simulation des autorisations. Après avoir confirmé la durée de l'accès et votre AWS identité, choisissez Approuver pour accorder l'accès. Si le fournisseur du produit demande un accès immédiat, il reçoit automatiquement des autorisations temporaires et la durée d'accès commence à courir. Sinon, informez la personne à l'origine de la demande de libérer l'accès au fournisseur du produit.

- Pour rejeter la demande, choisissez Refuser.
  - Dans la boîte de dialogue de rejet, indiquez clairement le motif du rejet afin d'aider le demandeur à comprendre pourquoi sa demande a été refusée.
  - Choisissez Refuser pour refuser l'accès.

8. La liste des demandes est automatiquement actualisée pour afficher les informations d'état les plus récentes. Vous pouvez également actualiser manuellement la page pour vérifier les mises à jour de statut.

Révoquer l'accès temporaire à une délégation

Bien que les sessions d'accès des fournisseurs de produits soient conçues pour expirer automatiquement après leur durée approuvée, il se peut que vous deviez mettre fin à l'accès immédiatement dans certaines situations. La révocation de l'accès actif du fournisseur de produits fournit un mécanisme de contrôle d'urgence lorsque des problèmes de sécurité surviennent, lorsque le travail du fournisseur de produits est terminé plus tôt que prévu ou lorsque les exigences commerciales changent. Les initiateurs de demandes et les administrateurs peuvent révoquer l'accès pour maintenir la sécurité et le contrôle opérationnel.

Pour révoquer l'accès temporaire à une délégation

1. Connectez-vous à la console de AWS gestion et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation de gauche, choisissez Demandes de délégation temporaires.
3. Localisez l'ID de demande pour la session d'accès que vous souhaitez révoquer.
4. Choisissez Actions, puis choisissez Révoquer l'accès.
5. Dans la boîte de dialogue, choisissez Révoquer l'accès pour confirmer que vous souhaitez mettre fin immédiatement à la session d'accès.

Une fois l'accès révoqué, le fournisseur du produit ne pourra plus accéder à vos AWS ressources. La révocation est enregistrée à des AWS CloudTrail fins d'audit.

Important

La révocation de l'accès met immédiatement fin à la session d'accès du fournisseur du produit. Tout travail ou processus en cours utilisant l'accès sera interrompu. Assurez-vous que la révocation ne perturbera pas les opérations critiques.

Note

Vous ne pouvez pas révoquer l'accès pour les demandes qui ont été approuvées par un utilisateur root. AWS vous recommande d'éviter d'utiliser un utilisateur root pour approuver les demandes de délégation. Utilisez plutôt un rôle IAM avec les autorisations appropriées.

Gestion des autorisations pour les demandes de délégation

Les administrateurs peuvent accorder aux responsables IAM des autorisations leur permettant de gérer les demandes de délégation émanant des fournisseurs de produits. Cela est utile lorsque vous souhaitez déléguer des pouvoirs d'approbation à des utilisateurs ou à des équipes spécifiques de votre organisation, ou lorsque vous devez contrôler qui peut effectuer des actions spécifiques sur les demandes de délégation.

Les autorisations IAM suivantes sont disponibles pour gérer les demandes de délégation :

Autorisation	Description
iam : Associate DelegationRequest	Associez une demande de délégation non attribuée à votre compte AWS
iam : GetDelega tionRequest	Afficher les détails d'une demande de délégation
iam : UpdateDel egationRequest	Transférer une demande de délégation à un administrateur pour approbation
iam : AcceptDel egationRequest	Approuver une demande de délégation

Autorisation	Description
iam : SendDelegationToken	Remettez le jeton d'échange au fournisseur du produit après approbation
iam : RejectDelegationRequest	Refuser une demande de délégation
iam : ListDelegationRequests	Répertorier les demandes de délégation pour votre compte

> **Note**
>
> Par défaut, les responsables IAM qui initient une demande de délégation sont automatiquement autorisés à gérer cette demande spécifique. Ils peuvent l'associer à leur compte, consulter les détails de la demande, rejeter une demande, la transmettre à un administrateur pour approbation, remettre le jeton d'échange au fournisseur du produit après approbation de l'administrateur et répertorier les demandes de délégation qui leur appartiennent.

Notifications

La délégation temporaire IAM s'intègre aux notifications AWS utilisateur pour vous aider à rester informé des modifications de l'état des demandes de délégation. Les notifications sont particulièrement utiles pour les administrateurs qui doivent examiner et approuver les demandes de délégation.

Grâce aux notifications AWS utilisateur, vous pouvez configurer les alertes à diffuser via plusieurs canaux, notamment le courrier électronique, Amazon Simple Notification Service (SNS), le AWS Chatbot pour Slack ou Microsoft Teams et l'Application Console Mobile. AWS Cela garantit que les bonnes personnes sont informées au bon moment, ce qui permet de répondre plus rapidement aux approbations en attente ou de prendre connaissance des modifications d'accès. Vous pouvez également personnaliser les événements qui déclenchent des notifications en fonction des besoins et des exigences de sécurité de votre organisation.

Événements de notification disponibles

Vous pouvez vous abonner pour recevoir des notifications concernant les événements de délégation temporaire IAM suivants :

- Demande de délégation temporaire IAM créée
- Demande de délégation temporaire IAM attribuée
- Demande de délégation temporaire IAM en attente d'approbation
- Demande de délégation temporaire IAM rejetée
- Demande de délégation temporaire IAM acceptée
- Demande de délégation temporaire IAM finalisée
- Demande de délégation temporaire IAM expirée

Configuration des notifications

Pour configurer les notifications pour les événements de délégation temporaire IAM :

1. Ouvrez la console de notifications AWS utilisateur
2. Création ou mise à jour d'une configuration de notification
3. Sélectionnez AWS IAM comme service
4. Choisissez les événements relatifs à la demande de délégation dont vous souhaitez être informé
5. Configurez vos canaux de diffusion (e-mail, AWS Chatbot, etc.)

Pour obtenir des instructions détaillées sur la configuration des notifications AWS utilisateur, y compris la configuration des canaux de diffusion et la gestion des règles de notification, consultez la documentation relative aux notifications AWS utilisateur.

CloudTrail

Toutes les actions effectuées par les fournisseurs de produits à l'aide d'un accès délégué temporaire sont automatiquement enregistrées AWS CloudTrail. Cela fournit une visibilité et une auditabilité complètes de l'activité des fournisseurs de produits sur votre AWS compte. Vous pouvez identifier quelles actions ont été entreprises par les fournisseurs de produits, à quel moment elles se sont produites et quel compte de fournisseur de produits les a effectuées.

Pour vous aider à faire la distinction entre les actions entreprises par vos propres responsables IAM et celles prises par les fournisseurs de produits dotés d'un accès délégué, les CloudTrail événements incluent un nouveau champ appelé `invokedByDelegate` sous l'`userIdentity`élément. Ce champ contient l'identifiant du AWS compte du fournisseur du produit, ce qui facilite le filtrage et l'audit de toutes les actions déléguées.

CloudTrail Structure de l'événement

L'exemple suivant montre un CloudTrail événement correspondant à une action exécutée par un fournisseur de produits à l'aide d'un accès délégué temporaire :

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:Role-Session-Name",
        "arn": "arn:aws:sts::111122223333:assumed-role/Role-Name/Role-Session-Name",
        "accountId": "111122223333",
        "accessKeyId": "[REDACTED:AWS_ACCESS_KEY]",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2024-09-09T17:50:16Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedByDelegate": {
            "accountId": "444455556666"
        }
    },
    "eventTime": "2024-09-09T17:51:44Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "GetUserPolicy",
    "awsRegion": "us-east-1",
    "requestParameters": {
        "userName": "ExampleIAMUserName",
        "policyName": "ExamplePolicyName"
```

```
    },
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

Le `invokedByDelegate` champ contient l'ID de AWS compte du fournisseur du produit qui a effectué l'action à l'aide de l'accès délégué. Dans cet exemple, le compte 444455556666 (le fournisseur du produit) a effectué une action sur le compte 111122223333 (le compte client).

Délégation temporaire de l'IAM pour AWS Partenaires

Présentation de

La délégation temporaire IAM permet aux AWS clients d' and/or intégrer facilement les produits AWS Partner dans leur AWS environnement grâce à des flux de travail interactifs et guidés. Les clients peuvent accorder aux AWS partenaires un accès temporaire limité pour configurer les AWS services requis, réduisant ainsi les difficultés liées à l'intégration et accélérant le délai de rentabilisation.

Pour intégrer la délégation temporaire à votre produit, apprenez à comprendre votre intégration, passez en revue la compréhension des autorisations et suivez les directives d'évaluation des politiques. Pour des exemples de politiques IAM, consultez les modèles ??? de politiques. Lorsque vous êtes prêt à implémenter, consultez la section Création de votre intégration.

La délégation temporaire IAM permet aux partenaires de :

- Simplifiez l'intégration des clients grâce au provisionnement automatique des ressources
- Réduisez la complexité de l'intégration en éliminant les étapes de configuration manuelles
- Instaurez la confiance grâce à des autorisations transparentes et approuvées par le client
- Permettez des opérations continues avec des modèles d'accès à long terme en utilisant des limites d'autorisation

Comment ça marche

1. Le partenaire crée une demande de délégation - Les partenaires créent une demande en spécifiant les autorisations dont ils ont besoin et pour combien de temps
2. Avis des clients dans AWS la console : le client voit exactement quelles autorisations le partenaire demande et pourquoi

3. Le client approuve - Le client approuve la demande et libère un jeton d'échange. Le jeton est envoyé au partenaire sur ce sujet SNS spécifié.
4. Le partenaire reçoit des informations d'identification temporaires - Les partenaires échangent le jeton contre des AWS informations d'identification temporaires
5. Le partenaire configure les ressources - Les partenaires utilisent les informations d'identification pour configurer les ressources requises dans le compte du client

Qualification des partenaires

Pour bénéficier de l'intégration de la délégation temporaire, un partenaire doit répondre aux exigences suivantes :

- Participation à ISV Accelerate — Vous devez être inscrit au programme ISV Accelerate (ISVA).
- AWS Mise en vente sur la place de marché : votre produit doit être répertorié AWS sur la place de marché avec un badge « Déployé sur AWS ».

Processus d'intégration

Procédez comme suit pour intégrer la délégation temporaire à votre produit :

1. Étape 1 : Réviser les exigences

   Consultez cette documentation pour comprendre les exigences de qualification et remplissez le questionnaire destiné aux partenaires ci-dessous.
2. Étape 2 : Soumettez votre demande d'intégration

   Envoyez un e-mail à aws-iam-partner-onboarding@amazon.com ou contactez votre AWS représentant. Incluez votre questionnaire à l'intention des partenaires rempli avec tous les champs obligatoires du tableau ci-dessous.
3. Étape 3 : AWS validation et révision

   AWS va :

   - Validez que vous répondez aux critères de qualification
   - Passez en revue vos modèles de politiques et vos limites d'autorisation
   - Donnez votre avis sur les artefacts que vous avez soumis

4. Étape 4 : Affinez vos politiques

   Répondez aux AWS commentaires et soumettez des modèles de politiques ou des limites d'autorisation mis à jour, selon les besoins.

5. Étape 5 : Compléter l'inscription

   Après approbation, AWS va :

   - Activez l'accès à l'API pour les comptes que vous avez spécifiés
   - Partagez les ARN pour votre modèle de politique et votre limite d'autorisations (le cas échéant)

   Vous recevrez une confirmation lorsque l'intégration sera terminée. Vous pouvez ensuite accéder à des API de délégation temporaires CreateDelegationRequest et, GetDelegatedAccessToken depuis vos comptes enregistrés, commencer à intégrer des flux de travail de demande de délégation dans votre produit.

Questionnaire pour partenaires

Le tableau suivant répertorie les informations requises pour l'intégration des partenaires :

Informations	Description	Obligatoire
ID de compte Partner Central	ID de compte de votre AWS compte enregistré sur AWS Partner Central.	Oui
PartnerId	Identifiant de partenaire fourni par AWS Partner Central.	Non
AWS Identifiant du produit Marketplace	ID de produit pour votre produit fourni par AWS Partner Central.	Oui
AWS Identifiants de compte	La liste de vos identifiants de AWS compte que vous souhaitez utiliser pour appeler des API de délégation temporaires. Cela devrait inclure à la fois vos comptes de production et les activités non liées à vos production/test comptes.	Oui

Informations	Description	Obligatoire
Nom du partenaire	Ce nom est affiché aux clients dans la console AWS de gestion lorsqu'ils examinent votre demande de délégation temporaire.	Oui
E-mail (s) de contact	Une ou plusieurs adresses e-mail pour vous contacter à propos de votre intégration.	Oui
Domaine du demandeur	Votre domaine (par exemple, www.exemple.com)	Oui
Description de l'intégration	Brève description du cas d'utilisation que vous souhaitez traiter à l'aide de cette fonctionnalité. Vous pouvez inclure des liens de référence vers votre documentation ou d'autres documents publics.	Oui
Diagramme d’architecture	Schéma d'architecture illustrant vos cas d'utilisation en matière d'intégration.	Non
Modèle de stratégie	Vous devez enregistrer au moins un modèle de politique pour cette fonctionnalité. Le modèle de politique définit les autorisations temporaires que vous souhaitez demander sur les AWS comptes des clients. Pour plus d'informations, consultez la section Modèle de politique.	Oui
Nom du modèle de politique	Nom du modèle de politique que vous souhaitez enregistrer.	Oui

Informations	Description	Obligatoire
Limite d'autorisations	Si vous souhaitez créer des rôles IAM dans les comptes clients à l'aide d'autorisations temporaires, vous devez enregistrer une limite d'autorisations auprès d'IAM. Des limites d'autorisation seront associées aux rôles IAM que vous créez afin de limiter le maximum d'autorisations sur le rôle. Vous pouvez utiliser certaines politiques AWS gérées comme limite d'autorisation ou enregistrer une nouvelle limite d'autorisation personnalisée (JSON). Pour plus d'informations, consultez la section Limite des autorisations.	Non
Nom de la limite d'autorisation	Le nom de votre limite d'autorisation. Le format est : arn:aws:iam : :partner :policy/permission_boundary/ <partner_domain>/<policy_name>_ <date>Le nom de la politique doit inclure la date de création comme suffixe. Le nom ne peut pas être mis à jour une fois la limite d'autorisation créée. Si vous utilisez une stratégie AWS gérée existante, indiquez plutôt l'ARN de la stratégie gérée.	Non
Description des limites d'autorisation	Description de la limite d'autorisation. Cette description ne peut pas être mise à jour une fois la limite d'autorisation créée.	Non

Comprendre votre intégration

Une fois le processus d'intégration terminé, vous pouvez développer votre intégration avec la délégation temporaire IAM. Une intégration complète implique généralement trois grandes catégories de travail :

1. Expérience utilisateur et conception des flux de travail

Créez une expérience frontale dans l'application partenaire qui guide les clients tout au long du flux de travail de délégation temporaire. La candidature d'un partenaire doit :

- Présentez un flux d'intégration ou de configuration clair dans lequel les clients peuvent accorder un accès temporaire. Étiquetez clairement cette action, par exemple « Déployer avec une délégation temporaire IAM ».
- Redirigez les clients vers la console de AWS gestion pour examiner et approuver la demande de délégation à l'aide du lien de console renvoyé par l' CreateDelegationRequest API
- Fournissez des messages appropriés sur les autorisations demandées et sur les raisons pour lesquelles elles sont demandées. Les clients peuvent voir ce message sur la page des détails de la demande de délégation.
- Gérez le retour du client vers votre demande une fois qu'il a terminé l'approbation dans AWS.

Meilleures pratiques pour les demandes de délégation temporaire

Lorsque vous implémentez la délégation temporaire pour votre application partenaire, suivez les bonnes pratiques définies ci-dessous pour aider les clients à vérifier l'authenticité et l'exactitude des demandes.

1. Inclure User-Identifiable le contexte dans les messages de demande

Vous devez inclure un contexte spécifique à l'utilisateur dans le message de demande de délégation. Ces informations aident les utilisateurs à identifier les demandes issues du flux de travail de leurs partenaires existants et à distinguer les demandes légitimes. Informations suggérées dans le message de demande :

- Identifiant de compte ou nom d'utilisateur du client dans votre service
- Nom de l'espace de travail, ID d'abonnement ou identifiant de l'organisation
- Ressource spécifique en cours d'accès (nom du cluster, nom du projet, environnement)
- Identifiant de transaction ou de demande unique généré pour cette tentative de délégation

Exemple de message de demande :

```
Request from Partner A workspace "production-analytics"
Account: user@example.com
Workspace ID: 1234ABCD
Cluster: ml-training-cluster-01
Request ID: 1111-2222-3333-4444
```

2. Facultatif : inclure l'identifiant AWS du compte lors du lancement de la délégation

Si l'identifiant du compte client est disponible, incluez-le dans la demande de délégation. Cette étape de vérification crée une liaison supplémentaire entre l'intention du client et le jeton de délégation.

3. Messages de demande de conception pour la vérification de sécurité

Structurez vos messages de demande pour permettre aux clients de vérifier en toute confiance la légitimité avant d'accorder l'accès. Exigences relatives à l'expérience utilisateur :

- Affichez le message de demande bien en évidence dans l'interface de votre application avant de rediriger vers AWS
- Utilisez un langage clair et descriptif qui est directement lié à l'action actuelle du client
- Évitez les messages génériques qui pourraient s'appliquer à toute demande de délégation
- Incluez suffisamment de détails pour que les clients puissent vérifier que la demande correspond au flux de travail prévu
- Afficher l'identifiant du AWS compte qui bénéficiera de l'accès délégué
- Fournissez une explication claire des autorisations demandées par votre application

Incluez un avis de confirmation dans votre demande avant de rediriger les clients vers l'écran de AWS consentement :

```
## Redirecting to IAM Temporary Delegation
You are about to grant [Your Service Name] temporary access to your AWS account.

Before clicking "Allow" on the AWS consent screen:
• Verify the request details match your current action
• Confirm the AWS account ID matches your intended account
• Make sure you initiated this request from [Your Service Name]
```

4. Recommandations contraignantes pour les sessions

Pendant qu' AWS IAM gère le flux d'autorisation de base, implémentez les pratiques suivantes dans votre application afin de renforcer l'intégrité des sessions :

- Générez des identifiants de demande uniques à usage unique pour chaque tentative de délégation

- Associez les demandes de délégation à la session d'application active du client
- Implémentez une expiration de demande appropriée en fonction de votre processus de délégation
- Validez que les rappels de délégation correspondent au contexte de la demande d'origine
- Consignez toutes les initiations et les finalisations de demandes de délégation à des fins de surveillance de la sécurité

En suivant ces bonnes pratiques, vous contribuez à créer une expérience de délégation temporaire plus sécurisée.

2. Intégration des API

Utilisez les API de délégation temporaire IAM pour envoyer et gérer les demandes de délégation. Une fois vos AWS comptes enregistrés, vous pouvez accéder aux API suivantes :

- IAM CreateDelegationRequest — Crée une demande de délégation pour le AWS compte d'un client. Cette API renvoie un lien de console vers lequel vous redirigez les clients pour qu'ils examinent et approuvent la demande.
- AWS STS GetDelegatedAccessToken— Récupère les AWS informations d'identification temporaires une fois qu'un client a approuvé votre demande de délégation. Utilisez ces informations d'identification pour effectuer des actions sur le compte du client.

Votre intégration doit gérer le cycle de vie complet des demandes de délégation, y compris la création de demandes, le suivi de leur statut et la récupération des informations d'identification temporaires une fois approuvées.

3. Configuration et orchestration des ressources

Après avoir obtenu des informations d'identification temporaires, orchestrez les flux de travail nécessaires pour configurer les ressources du AWS compte du client. Cela pourrait inclure :

- Appelez directement les API des AWS services pour créer et configurer des ressources
- Déploiement de l'infrastructure à l' AWS CloudFormation aide
- Création de rôles IAM pour un accès continu (nécessite l'utilisation de limites d'autorisation). Pour vous défendre en profondeur, nous vous recommandons d'utiliser des noms de rôles dynamiques, par exemple un nom avec un suffixe généré de manière aléatoire. Cette approche rend le nom du rôle dans chaque compte client unique et imprévisible.

Votre logique d'orchestration doit être idempotente et gérer les défaillances avec élégance, car les clients peuvent avoir besoin de réessayer ou de modifier leurs approbations de délégation.

Comprendre les autorisations

Dans le cadre du processus d'intégration des fonctionnalités, vous devrez enregistrer auprès d'IAM des politiques qui définissent les autorisations que vous souhaitez demander sur les comptes des clients AWS . Le processus d'enregistrement offre une expérience plus cohérente aux clients et permet d'éviter les pièges courants lors de la rédaction de politiques.

Lors de l'inscription, AWS évalue vos politiques par rapport à un ensemble de validations. Ces validations visent à normaliser le formatage et la structure des politiques et à fournir des protections de base contre les anti-modèles connus. Les validations réduisent également le risque d'augmentation des privilèges, d'accès involontaire entre comptes et d'accès étendu à des ressources de grande valeur dans les comptes clients.

Types d'autorisations

AWS prendra en compte deux catégories d'autorisations : temporaires et à long terme.

Autorisations temporaires

Les autorisations temporaires limitent les autorisations attribuées à toutes les sessions d'accès délégué temporaires. Les autorisations temporaires sont décrites dans les modèles de politique appliqués à la session déléguée. Les modèles prennent en charge les paramètres que vous fournissez lorsque vous créez une demande de délégation. Ces valeurs de paramètres sont ensuite liées à la session. Les autorisations temporaires fonctionnent de la même manière que les politiques de session disponibles AWS STS aujourd'hui : les politiques limitent les capacités de l'utilisateur sous-jacent, mais n'accordent aucun accès supplémentaire. Pour plus d'informations, consultez la AWS STS documentation sur les politiques de session.

Long-Term Autorisations

Long-term les autorisations limitent les autorisations de tous les rôles créés ou gérés via un accès temporaire. Long-term les autorisations sont implémentées en tant que limites d'autorisations IAM. Vous pouvez soumettre une ou plusieurs limites d'autorisation dans AWS le cadre de l'intégration. Après approbation, AWS nous partagerons avec vous un ARN de politique que vous pourrez référencer dans vos politiques.

Ces politiques de délimitation présentent deux caractéristiques remarquables. Tout d'abord, ils sont immuables. Si vous souhaitez mettre à jour les autorisations, vous pouvez enregistrer une nouvelle

limite d'autorisations. Vous pouvez ensuite associer la nouvelle limite d'autorisations aux rôles de vos clients en envoyant une nouvelle demande de délégation. Deuxièmement, les politiques ne sont pas modélisées. Étant donné que la même politique de limites est partagée à l'échelle mondiale, elle ne peut pas être modifiée par client.

> **Important**
>
> Les limites d'autorisation ont une taille maximale de 6 144 caractères.

> **Note**
>
> Si vous souhaitez mettre à jour une limite d'autorisation ou un modèle de politique, contactez IAM à l'adresse aws-iam-partner-onboarding@amazon.com. Une fois la nouvelle limite d'autorisation enregistrée, vous pouvez envoyer une demande de délégation aux clients pour mettre à jour le rôle IAM et associer la limite d'autorisation nouvellement enregistrée. Consultez la section Exemples pour plus de détails.

Exemple de cas d'utilisation : charge de travail liée au traitement des données

Prenons l'exemple d'un fournisseur de produits qui gère une charge de travail de traitement des données dans les comptes clients. Le fournisseur doit mettre en place une infrastructure lors de l'intégration initiale, mais a également besoin d'un accès continu pour gérer la charge de travail.

Autorisations temporaires (pour la configuration initiale) :

- Création d'instances Amazon EC2, de VPC et de groupes de sécurité
- Création d'un compartiment Amazon S3 pour les données traitées
- Création d'un rôle IAM pour les opérations en cours
- Associer une limite d'autorisation au rôle IAM

Long-term autorisations (rôle IAM avec limite d'autorisation pour les opérations en cours) :

- Démarrez et arrêtez des instances Amazon EC2 pour exécuter des tâches de traitement
- Lire les données d'entrée depuis le compartiment Amazon S3
- Écrire les résultats traités dans le compartiment Amazon S3

Les autorisations temporaires sont utilisées une seule fois lors de l'intégration pour configurer l'infrastructure. Le rôle IAM créé au cours de ce processus possède une limite d'autorisations qui limite ses autorisations maximales aux seules opérations nécessaires à la gestion continue de la charge de travail. Cela garantit que même si les politiques du rôle sont modifiées, il ne peut pas dépasser les autorisations définies dans la limite.

Directives d'évaluation des politiques

AWS évaluera les politiques que vous avez soumises par rapport à un ensemble de directives. Les mêmes directives d'évaluation s'appliquent à la fois aux modèles de politique et aux limites d'autorisation, avec des différences mineures notées le cas échéant.

Aux fins de l'évaluation, les services sont séparés en groupes distincts. La distinction la plus importante concerne les services sensibles à la sécurité, qui gèrent l'accès, les informations d'identification et les clés. Les politiques donnant accès à ces services doivent être étroitement axées sur le travail en cours. Security-sensitive les services incluent les suivants : gestion des AWS identités et des accès (IAM), service de gestion des AWS clés (KMS), gestionnaire d'accès aux AWS ressources (RAM), centre d'identité AWS IAM, AWS organisations et gestionnaire de AWS secrets.

Une distinction secondaire concerne les services qui peuvent accéder aux données au-delà des limites des comptes. Les politiques relatives à ces services doivent inclure des protections visant à empêcher tout accès involontaire entre comptes.

Validations courantes

Toutes les déclarations de politique doivent suivre les directives suivantes :

- Toutes les instructions doivent inclure les champs Effet, Action (ou NotAction), Ressource et Condition dans cet ordre
- Toutes les actions d'une même instruction doivent être répertoriées par ordre alphabétique
- Tous les ARN inclus dans la politique doivent suivre la syntaxe définie dans la documentation publique pour les services concernés
- NotAction les champs ne peuvent être utilisés que dans les instructions Deny
- Les actions figurant dans les instructions Allow doivent inclure un code de service. Les caractères génériques (« * ») ne sont pas autorisés

Security-sensitive restrictions de service

Les restrictions suivantes s'appliquent aux services sensibles en matière de sécurité mentionnés ci-dessus :

- Les actions des instructions Allow doivent être plus spécifiques que celles de [service] :*
- Les actions figurant dans les instructions Allow pour les modèles de politique d'accès temporaire ne doivent pas contenir de caractères génériques
- Les actions sensibles, telles que iam : PassRole ou iam :CreateServiceLinkedRole, nécessitent une portée supplémentaire, telle que des ressources spécifiques ou des vérifications conditionnelles. Ces actions incluent :
  - Transfert de rôle IAM
  - Actions de modification des rôles IAM
  - Actions de modification de la politique IAM
  - AWS Opérations d'écriture ou de chiffrement KMS
  - AWS Opérations d'écriture ou de partage de RAM
  - AWS Opérations de Secrets Manager pour récupérer ou modifier des secrets, ou modifier des politiques relatives aux ressources
- D'autres actions peuvent utiliser une ressource générique, telle que iam : ListUsers ou iam : GetPolicy
- Les actions qui gèrent les informations d'identification, telles que iam :CreateAccessKey, sont bloquées

IAM-specific restrictions

Pour IAM :

- Seules des opérations d'écriture limitées sont autorisées pour les rôles et les politiques IAM. Vous ne pouvez pas demander d'autorisations sur d'autres ressources IAM telles que les utilisateurs, les groupes et les certificats.
- Les actions de saisie de règles ou de gestion de stratégie en ligne sont limitées aux rôles dotés d'une limite d'autorisations. Les limites d'autorisation doivent être fournies par le partenaire ou figurer sur une liste de politiques AWS gérées autorisées. AWS les politiques gérées peuvent être autorisées si elles n'accordent pas d'autorisations hautement privilégiées ou administratives. Par exemple, les politiques AWS gérées pour des fonctions spécifiques ou la SecurityAudit politique

peuvent être acceptables. AWS examinera chaque politique AWS gérée au cas par cas lors du processus d'intégration.

- La gestion des politiques n'est autorisée que pour les politiques ayant un chemin spécifique au partenaire : arn:aws:iam : :@ {} : .com/ [feature] * AccountId policy/partner_domain
- Les balises ne peuvent être appliquées que lors de la création de la ressource, et uniquement pour les rôles et les politiques
- iam : PassRole les contrôles doivent correspondre à un nom ou à un préfixe de chemin spécifique

AWS STS-restrictions spécifiques

Pour AWS STS :

- sts : AssumeRole doit être limité à un ARN de rôle spécifique, à un préfixe d'ARN de rôle, ou limité à un ensemble de comptes ou à une unité organisationnelle ID/organizational

Restrictions relatives à l'IAM Identity Center

Pour AWS IAM Identity Center, les actions suivantes sont bloquées :

- Toutes les actions relatives à la gestion des autorisations (par exemple, sso :AttachCustomerManagedPolicyReferenceToPermissionSet)
- Modifications des utilisateurs, des groupes et des membres pour AWS Identity Store
- Gestion des balises

AWS Restrictions des organisations

Pour AWS les organisations, seules les actions de lecture seront autorisées.

Validations supplémentaires spécifiques aux services

- Les actions qui obtiennent des secrets ou des informations d'identification, telles que glue : GetConnection ou redshift :GetClusterCredentials, doivent avoir des conditions correspondant à des ARN complets, à des préfixes ARN ou à des balises
- Pour Amazon Redshift : redshift : n'GetClusterCredentials est autorisé que sur un nom de base de données spécifique, et redshift : n'GetClusterCredentialsWithIAM est autorisé que sur un nom de groupe de travail spécifique

> **Note**
>
> Lorsque vous gérez les ressources IAM du compte, nous vous recommandons d'utiliser un chemin indiquant votre nom, par exemple arn:aws:iam : :111122223333 :. role/partner com/ rolename. Cela permettra de différencier les ressources associées à votre intégration et de faciliter la découverte, l'audit et l'analyse pour les clients.

Cross-account exigences d'accès

Les relevés susceptibles d'autoriser l'accès entre comptes doivent inclure au moins l'un des éléments suivants :

- Une condition spécifiant le compte ou l'organisation de la ressource (par exemple, aws : ResourceOrgId correspondance d'une ou de plusieurs valeurs attendues)
- Un champ de ressource qui inclut un compte spécifique (par exemple, arn:aws:sqs : *:111122223333 : *)
- Un champ de ressource qui inclut un compte non générique et un nom de ressource complet (par exemple, arn:aws:s3 : ::full-bucket-name)

> **Note**
>
> Cross-account l'accès est une fonctionnalité sensible qui nécessite une justification commerciale claire. AWS étudiera attentivement la nécessité d'un accès multicompte lors du processus d'intégration.

Modèles de politique

Les modèles de politiques sont une nouvelle structure IAM conçue pour définir les autorisations temporaires que les partenaires demandent sur les comptes des clients. Comme les politiques IAM classiques, elles définissent les autorisations à l'aide d'instructions comportant des éléments Effect, Action, Resource et Condition. La principale différence réside dans le fait que les modèles de politique incluent des paramètres (tels que @ {bucketName}) qui sont remplacés par des valeurs réelles lorsque vous créez une demande de délégation.

Comment fonctionnent les modèles de politiques

Dans le cadre du processus d'intégration, vous enregistrez vos modèles de politique auprès AWS de. AWS attribue à chaque modèle un ARN unique auquel vous faites référence lorsque vous créez des demandes de délégation.

Lorsque vous créez une demande de délégation, vous spécifiez :

- L'ARN du modèle de politique
- Valeurs de paramètres à substituer dans le modèle

AWS combine le modèle avec les valeurs de vos paramètres pour générer une politique IAM standard. Les clients examinent cette politique finale lorsqu'ils approuvent votre demande de délégation, afin de savoir exactement quelles autorisations seront accordées.

> **Note**
>
> La taille maximale de la politique rendue finale est de 2 048 caractères.

Voici un exemple simple illustrant le fonctionnement de la substitution de modèles.

Modèle de politique :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::@{bucketName}/*"
        }
    ]
}
```

Paramètres fournis dans la demande de délégation :

```
{
```

```
    "Name": "bucketName",
    "Values": ["customer-data-bucket"],
    "Type": "String"
}
```

Politique finale (ce que les clients voient) :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::customer-data-bucket/*"
        }
    ]
}
```

Syntaxe du modèle

Les modèles de politiques utilisent deux fonctionnalités clés pour apporter de la flexibilité : la substitution de paramètres et les instructions conditionnelles. La substitution de paramètres vous permet de définir des espaces réservés dans votre modèle qui sont remplacés par des valeurs réelles lors de la création d'une demande de délégation. Les instructions conditionnelles vous permettent d'inclure ou d'exclure des déclarations de stratégie complètes en fonction des valeurs des paramètres.

Substitution et types de paramètres

Utilisez la syntaxe @ {parameterName} pour définir les paramètres de votre modèle de politique. Lorsque vous créez une demande de délégation, vous devez spécifier le type de chaque paramètre.

String

Une valeur unique qui est substituée directement dans le modèle.

Modèle :

```
"Resource": "arn:aws:s3:::@{bucketName}/*"
```

Paramètres :

```
{
    "Name": "bucketName",
    "Values": ["my-bucket"],
    "Type": "String"
}
```

Résultat rendu :

```
"Resource": "arn:aws:s3:::my-bucket/*"
```

StringList

Plusieurs valeurs qui génèrent plusieurs entrées de ressources. Lorsqu'un StringList paramètre est utilisé dans l'ARN d'une ressource, il se développe pour créer des entrées de ressource distinctes pour chaque valeur.

Modèle :

```
"Resource": "arn:aws:s3:::@{bucketNames}/*"
```

Paramètres :

```
{
    "Name": "bucketNames",
    "Values": ["bucket-1", "bucket-2"],
    "Type": "StringList"
}
```

Résultat rendu :

```
"Resource": [
    "arn:aws:s3:::bucket-1/*",
    "arn:aws:s3:::bucket-2/*"
]
```

Cross-Product Comportement

Lorsque plusieurs paramètres sont utilisés dans le même ARN de ressource, StringList les paramètres créent un produit croisé de toutes les combinaisons.

Modèle :

```
"Resource": "arn:aws:s3:::@{bucketNames}/@{prefix}/*"
```

Paramètres :

```
[
    {
        "Name": "bucketNames",
        "Values": ["bucket-1", "bucket-2"],
        "Type": "StringList"
    },
    {
        "Name": "prefix",
        "Values": ["data"],
        "Type": "String"
    }
]
```

Résultat rendu :

```
"Resource": [
    "arn:aws:s3:::bucket-1/data/*",
    "arn:aws:s3:::bucket-2/data/*"
]
```

Déclarations conditionnelles

Utilisez la directive @Enabled pour inclure ou exclure de manière conditionnelle des instructions entières en fonction des valeurs des paramètres.

Syntaxe :

- @Enabled : « ParameterName » - Inclut l'instruction lorsque la valeur du paramètre est « True »
- @Enabled : « ! ParameterName » - Inclut l'instruction lorsque la valeur du paramètre n'est PAS « vraie » (négation)

Modèle :

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["s3:GetObject"],
            "Resource": "*"
        },
        {
            "@Enabled": "ENABLE_S3_WRITE",
            "Effect": "Allow",
            "Action": ["s3:PutObject"],
            "Resource": "arn:aws:s3:::@{bucketName}/*"
        }
    ]
}
```

Paramètres (lorsque ENABLE_S3_WRITE est « True ») :

```
[
    {
        "Name": "bucketName",
        "Values": ["my-bucket"],
        "Type": "String"
    },
    {
        "Name": "ENABLE_S3_WRITE",
        "Values": ["True"],
        "Type": "String"
    }
]
```

Résultat rendu :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["s3:GetObject"],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": ["s3:PutObject"],
```

```
            "Resource": "arn:aws:s3:::my-bucket/*"
        }
    ]
}
```

Paramètres (lorsque ENABLE_S3_WRITE vaut « False ») :

```
[
    {
        "Name": "bucketName",
        "Values": ["my-bucket"],
        "Type": "String"
    },
    {
        "Name": "ENABLE_S3_WRITE",
        "Values": ["False"],
        "Type": "String"
    }
]
```

Résultat rendu :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["s3:GetObject"],
            "Resource": "*"
        }
    ]
}
```

Lorsque ENABLE_S3_WRITE est défini sur « True », l'instruction conditionnelle est incluse. Lorsqu'elle est définie sur « False », l'instruction est exclue de la politique rendue.

Exemples supplémentaires

Les exemples suivants illustrent des modèles courants d'utilisation de modèles de politiques dans le cadre de la délégation temporaire. Ils se concentrent sur la création de rôles IAM avec des limites d'autorisation pour un accès à long terme et présentent différentes stratégies pour définir la portée des autorisations pour des ressources spécifiques. Ces exemples montrent comment trouver un

équilibre entre flexibilité et sécurité en utilisant des techniques telles que les préfixes ARN, le balisage des ressources et la mise à jour des limites d'autorisation.

Exemple 1 : accorder l' Long-Term accès à des ressources spécifiques

La limite d'autorisation suivante est soumise sous la forme SQSAccessorBoundary « » pour « partner.com » :

```
{
    "Effect": "Allow",
    "Action": [
        "sqs:DeleteMessage",
        "sqs:ReceiveMessage",
        "sqs:SendMessage"
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
}
```

> **Note**
>
> Cela inclut une condition relative au même compte afin d'éviter d'accorder l'accès aux files d'attente d'autres comptes dotés de politiques de ressources ouvertes. Une référence directe à l'identifiant de compte du client ne peut pas être incluse car la limite est partagée par tous les clients et ne peut pas être modélisée.

Comme il s'agit de la première version de cette politique, son ARN est arn:aws:iam : :partner : -. policy/permissions boundary/partner com/SQSAccessorBoundary_2025_01_15

Le modèle de politique suivant est soumis pour les autorisations d'accès temporaires :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "sqs:ListQueues"
            ],
            "Resource": "arn:aws:sqs:*:*:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateRole",
                "iam:PutRolePermissionsBoundary",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::@{AccountId}:role/partner.com/SQSAccessor",
            "Condition": {
                "StringEquals": {
                    "iam:PermissionsBoundary": "arn:aws:iam::partner:policy/
permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_15"
                }
            }
        }
    ]
}
```

Exemple 2 : Utilisation de préfixes ARN

La limite d'autorisation peut spécifier un préfixe ARN de ressource pour limiter l'accès :

```
"Resource": "arn:aws:sqs:*:@{AccountId}:PartnerPrefix*"
```

Cela limite l'accès aux seules ressources portant ce préfixe, réduisant ainsi la portée des ressources accessibles.

Exemple 3 : Utilisation de balises pour le contrôle d'accès aux ressources

Vous pouvez baliser des ressources lors d'un accès délégué temporaire et vous fier à ces balises pour un contrôle d'accès à long terme.

Limite d'autorisation autorisant l'accès aux ressources balisées :

```
{
    "Effect": "Allow",
    "Action": [
        "sqs:DeleteMessage",
```

```
        "sqs:ReceiveMessage",
        "sqs:SendMessage"
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "Null": {
            "aws:ResourceTag/ManagedByPartnerDotCom": "false"
        },
        "StringEquals": {
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
}
```

Modèle de politique pour baliser les nouvelles files d'attente lors de leur création :

```
{
    "Effect": "Allow",
    "Action": [
        "sqs:CreateQueue",
        "sqs:TagQueue"
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "Null": {
            "aws:RequestTag/ManagedByPartnerDotCom": "false"
        }
    }
}
```

Modèle de politique pour baliser les files d'attente préexistantes et créer le rôle :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sqs:TagQueue"
            ],
            "Resource": "arn:aws:sqs:*:@{AccountId}:@{QueueName}",
            "Condition": {
                "ForAllValues:StringEquals": {
```

```
                    "aws:TagKeys": "ManagedByPartnerDotCom"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateRole",
                "iam:PutRolePermissionsBoundary",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::@{AccountId}:role/partner.com/SQSAccessor",
            "Condition": {
                "StringEquals": {
                    "iam:PermissionsBoundary": "arn:aws:iam::partner:policy/
permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_15"
                }
            }
        }
    ]
}
```

Cette approche permet aux clients de confirmer explicitement quelles ressources spécifiques sont accessibles à long terme.

Exemple 4 : mise à jour de la limite d'autorisation

Pour mettre à jour une limite d'autorisation, enregistrez une nouvelle version avec un nouveau suffixe de date et demandez l'autorisation de le remplacer.

Limite d'autorisation mise à jour avec autorisation supplémentaire :

```
{
    "Effect": "Allow",
    "Action": [
        "sqs:DeleteMessage",
        "sqs:PurgeQueue",
        "sqs:ReceiveMessage",
        "sqs:SendMessage"
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "StringEquals": {
```

```
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
}
```

En tant que deuxième version, cette politique possède l'ARN : arn:aws:iam : :partner : -. policy/permissions boundary/partner com/SQSAccessorBoundary_2025_01_20

Modèle de politique pour mettre à jour la limite d'autorisation du rôle existant :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:PutRolePermissionsBoundary"
            ],
            "Resource": "arn:aws:iam::@{AccountId}:role/partner.com/SQSAccessor",
            "Condition": {
                "StringEquals": {
                    "iam:PermissionsBoundary": "arn:aws:iam::partner:policy/
permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_20"
                }
            }
        }
    ]
}
```

Les clients doivent approuver cette demande de délégation pour mettre à jour la limite d'autorisation du rôle existant.

Construisez votre intégration

Comprendre le cycle de vie des demandes

Avant de créer votre intégration, il est important de comprendre comment les demandes de délégation progressent, de leur création à leur achèvement.

États de la demande

Une demande de délégation passe par les états suivants :

State	Description
Non assigné	Demande créée mais pas encore associée à un compte client et à un responsable IAM. La demande a peut-être été créée sans spécifier de compte cible, ou avec un identifiant de compte cible mais n'a pas encore été revendiquée par le propriétaire du compte.
Assigné	Demande associée à un compte client et en attente d'examen
En attente d'approbation	Le client a transmis la demande à un administrateur pour approbation
Acceptée	Demande approuvée par le client mais le jeton d'échange n'a pas encore été publié
Finalisé	Jeton d'échange remis au fournisseur du produit. La période de délégation (validité du jeton d'échange) commence lorsque la demande atteint l'état Finalisé
Refusée	Demande rejetée par le client
Expiré	La demande a expiré en raison d'une inactivité ou d'un délai d'attente

Transitions d'état

Flux normal (chemin d'approbation)

- Non attribué → Assigné : le client associe la demande à son compte
- Assigné → Accepté OU Assigné → En attente d'approbation : le client approuve la demande directement OU la transmet à l'administrateur pour examen
- En attente d'approbation → Acceptée : l'administrateur approuve la demande
- Accepté → Finalisé : le client libère le jeton d'échange

Voie de rejet

- Assigné → Rejeté : le client rejette la demande
- En attente d'approbation → Rejeté : l'administrateur rejette la demande

- Accepté → Rejeté : le client révoque l'approbation avant de libérer le jeton

Chemin d'expiration

Les demandes expirent automatiquement si aucune action n'est entreprise dans le délai spécifié :

- Non attribué → Expiré (1 jour)
- Attribué → Expiré (7 jours)
- En attente d'approbation → Expiré (7 jours)
- Accepté → Expiré (7 jours)
- Rejeté → Expiré (7 jours)
- Finalisé → Expiré (7 jours)

États des terminaux

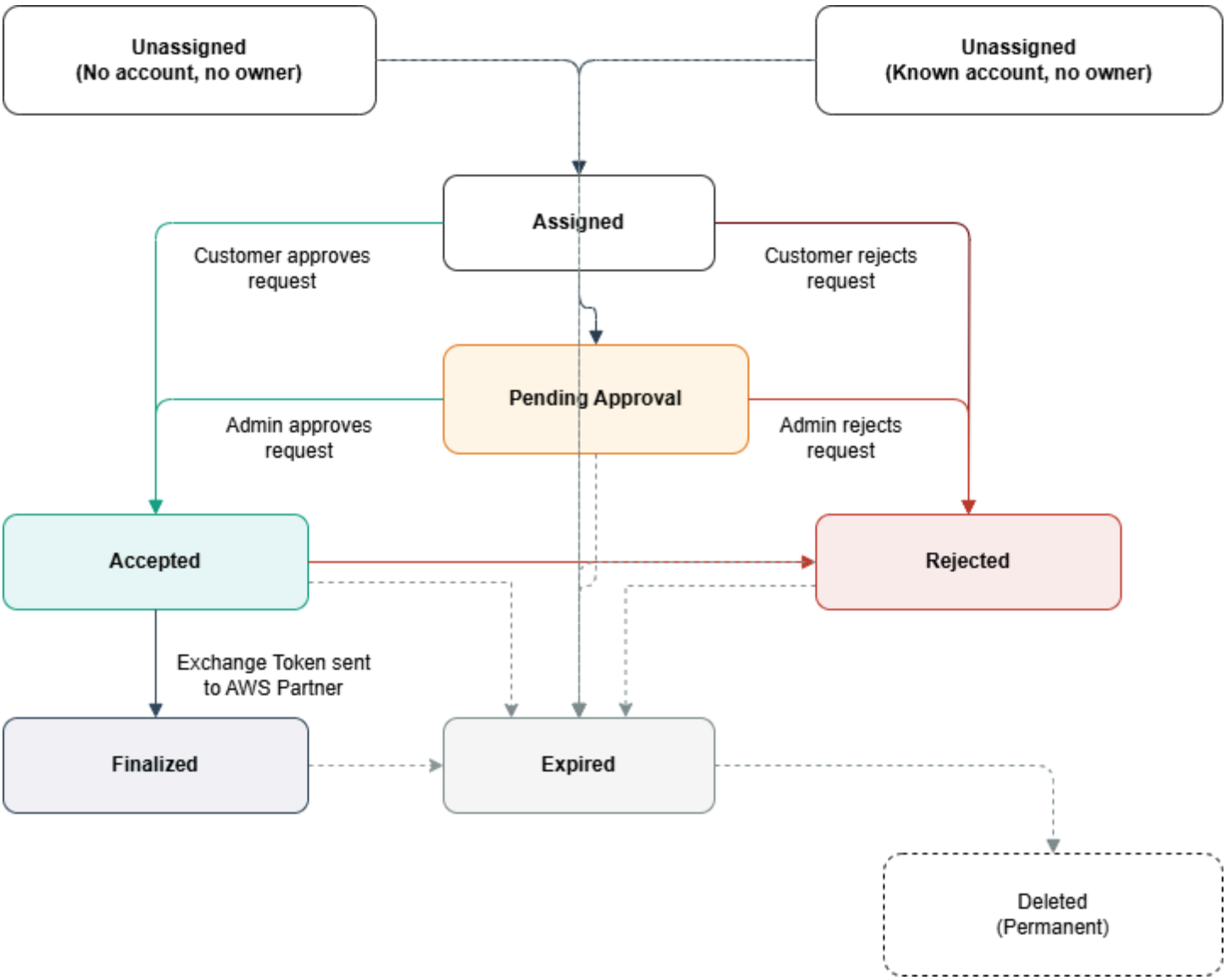
Les états suivants sont terminaux (aucune autre transition) :

- Finalisé : jeton d'échange envoyé
- Rejeté : la demande a été refusée
- Expiré : la demande a expiré ou la période de délégation est terminée

Les demandes expirées sont finalement supprimées du système après la période de conservation.

Gestion des états des demandes de délégation dans votre application

En tant que partenaire, vous devez suivre l'état des demandes de délégation dans votre système et les communiquer à vos clients. Lorsque vous recevez des notifications SNS concernant des changements d'état, stockez ces mises à jour dans votre backend et répercutez-les dans votre interface utilisateur orientée client. Portez une attention particulière à l'état En attente d'approbation : lorsqu'un client transmet une demande à un administrateur pour examen, il vous AWS envoie une notification en attente d'approbation. Les demandes peuvent rester dans cet état jusqu'à 7 jours en attendant l'action de l'administrateur. Pendant ce temps, montrez aux clients que leur demande est en attente d'approbation par l'administrateur dans votre application. Envisagez de fournir un lien profond vers la AWS console où les clients peuvent vérifier l'état de la demande ou contacter leur administrateur. Il est important de gérer correctement la machine d'état dans votre backend et de présenter les informations d'état correctes aux clients à chaque étape pour une bonne expérience d'intégration.



Configuration des notifications

IAM utilise Amazon Simple Notification Service (SNS) pour vous communiquer les modifications d'état des demandes de délégation. Lorsque vous créez une demande de délégation, vous devez fournir un ARN de sujet SNS à partir de votre AWS compte enregistré. IAM publiera des messages sur ce sujet lors d'événements importants, notamment lorsque les clients approuvent ou rejettent des demandes et lorsque le jeton d'échange est prêt.

> **Note**
>
> Les sujets SNS ne peuvent pas figurer dans les régions optionnelles. AWS Votre rubrique SNS doit se trouver dans une AWS région activée par défaut. Pour obtenir la liste des régions

optionnelles, consultez la section Gestion des AWS régions dans le guide de gestion des AWS comptes.

Configuration des rubriques SNS

Pour recevoir les notifications de demande de délégation, vous devez configurer votre rubrique SNS de manière à accorder à IAM les autorisations nécessaires pour publier des messages. Ajoutez la déclaration de politique suivante à votre politique en matière de sujets SNS :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIAMServiceToPublish",
            "Effect": "Allow",
            "Principal": {
                "Service": "iam.amazonaws.com"
            },
            "Action": "SNS:Publish",
            "Resource": "arn:aws:sns:us-east-1:111122223333:TOPIC-NAME"
        }
    ]
}
```

Important

La rubrique SNS doit se trouver dans l'un de vos AWS comptes enregistrés. IAM n'acceptera pas les sujets SNS provenant d'autres comptes. Si la politique thématique n'est pas correctement configurée, vous ne recevrez pas de notifications de changement d'état ni le jeton d'échange.

Types de notifications

IAM envoie deux types de notifications :

StateChange Notifications

Envoyé lorsqu'une demande de délégation passe à un nouvel état (attribuée, en attente d'approbation, acceptée, finalisée, rejetée, expirée).

ExchangeToken Notifications

Envoyé lorsqu'un client libère le jeton de délégation (état Finalisé). Cette notification inclut le jeton d'échange dont vous avez besoin pour obtenir des informations d'identification.

États de notification

Vous recevrez des notifications pour les états de demande de délégation suivants :

State	Type de notification	Description
ASSIGNÉ	StateChange	La demande a été associée à un compte client
EN ATTENTE D'APPROBATION	StateChange	Le client a transmis la demande à un administrateur pour approbation
ACCEPTÉ	StateChange	Le client a approuvé la demande mais n'a pas encore libéré le jeton
FINALISÉ	StateChange	Le client a publié le jeton d'échange
FINALISÉ	ExchangeToken	Cette notification contient le jeton d'échange
REFUSÉE	StateChange	Le client a rejeté la demande
EXPIRÉ	StateChange	La demande a expiré avant d'être terminée

Format du message de notification

IAM publie des notifications SNS standard. Les informations relatives à la demande de délégation sont contenues dans le champ Message sous forme de chaîne JSON.

Champs communs (toutes les notifications)

Champ	Type	Description
Type	Chaîne	Soit « StateChange » soit « ExchangeToken »

Champ	Type	Description
RequestId	Chaîne	L'ID de demande de délégation IAM
Requestor WorkflowId	Chaîne	L'ID de flux de travail que vous avez fourni lors de la création de la demande
State	Chaîne	État actuel de la demande
OwnerAccountId	Chaîne	Numéro de AWS compte du client
UpdatedAt	Chaîne	Horodatage du changement d'état (format ISO 8601)

Champs supplémentaires (ExchangeToken notifications uniquement)

Champ	Type	Description
ExchangeToken	Chaîne	Le jeton à échanger contre les informations d'identif ication à l'aide de l' AWS STS GetDelegatedAccessToken API
ExpiresAt	Chaîne	Quand l'accès délégué expire (format ISO 8601)

Exemples de notifications

StateChange Notification

```
{
  "Type": "Notification",
  "MessageId": "61ee8ad4-6eec-56b5-8f3d-eba57556aa13",
  "TopicArn": "arn:aws:sns:us-east-1:123456789012:partner-notifications",
  "Message": "{\"RequestorWorkflowId\":\"workflow-12345\",\"Type\":\"StateChange\",
\"RequestId\":\"dr-abc123\",\"State\":\"ACCEPTED\",\"OwnerAccountId\":\"111122223333\",
\"UpdatedAt\":\"2025-01-15T10:30:00.123Z\"}",
  "Timestamp": "2025-01-15T10:30:00.456Z",
  "SignatureVersion": "1",
  "Signature": "...",
  "SigningCertURL": "...",
  "UnsubscribeURL": "..."
```

```
}
```

ExchangeToken Notification

```
{
  "Type": "Notification",
  "MessageId": "e44e5435-c72c-5333-aba3-354406782f5b",
  "TopicArn": "arn:aws:sns:us-east-1:123456789012:partner-notifications",
  "Message": "{\"RequestId\":\"dr-abc123\",\"RequestorWorkflowId\":\"workflow-12345\",
\"State\":\"FINALIZED\",\"OwnerAccountId\":\"111122223333\",\"ExchangeToken\":
\"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...\",\"ExpiresAt\":\"2025-01-15T18:30:00.123Z\",
\"UpdatedAt\":\"2025-01-15T10:30:00.456Z\",\"Type\":\"ExchangeToken\"}",
  "Timestamp": "2025-01-15T10:30:00.789Z",
  "SignatureVersion": "1",
  "Signature": "...",
  "SigningCertURL": "...",
  "UnsubscribeURL": "..."
}
```

Jetons d'échange

Un jeton d'échange ou un jeton d'échange est émis par IAM lorsqu'un client accepte et finalise une demande de délégation. Le fournisseur du produit utilise ce jeton d'échange ou d'échange pour appeler l' AWS STS GetDelegatedAccessToken API afin d'obtenir des AWS informations d'identification temporaires avec les autorisations approuvées par les clients. Le jeton d'échange lui-même ne donne pas accès à vos AWS ressources ; il doit être échangé contre des informations d'identification réelles via AWS STS.

Le jeton d'échange ne peut être utilisé que par le compte du fournisseur du produit qui a créé la demande de délégation. Le compte demandeur est intégré au jeton, ce qui garantit que seul le fournisseur de produit autorisé peut obtenir des informations d'identification pour accéder au compte client.

Durée d'accès

La période de délégation commence lorsque le client libère le jeton d'échange, et non lorsque le fournisseur du produit l'échange. Une fois que le client a libéré le jeton :

- Le fournisseur du produit reçoit le jeton par le biais d'une notification SNS
- Ils peuvent immédiatement l'échanger contre des informations d'identification

- Les informations d'identification expirent à : heure de publication + durée approuvée
- Le fournisseur du produit peut échanger le jeton plusieurs fois avant son expiration pour obtenir de nouvelles informations d'identification si nécessaire

Rédemptions multiples

Les fournisseurs de produits peuvent échanger le jeton plusieurs fois pendant la période de validité pour obtenir de nouvelles informations d'identification. Cependant, toutes les informations d'identification obtenues à partir du même jeton d'échange expirent en même temps, en fonction de la date à laquelle vous avez publié le jeton.

Exemple : si vous approuvez une demande de délégation de 2 heures et que vous libérez le jeton à 10 h 00 :

Heure de sortie du jeton	Heure d'échange de jetons	Expiration des informations d'identification	Temps utilisable
10 h 00	10 h 00	12:00	2 heures
10 h 00	10 h 20	12:00	1 heure 40 minutes
10 h 00	11 h 40	12:00	20 minutes
10 h 00	12 h 10	Échec (le jeton a expiré)	0 minutes

Comme le montre le tableau, l'échange du jeton plus tard dans la période de validité réduit le temps d'utilisation pour le fournisseur du produit.

Gestionnaire d'accès au compte

Qu'est-ce que le gestionnaire d'accès au compte ?

Le gestionnaire d'accès aux comptes vous permet Compte AWS d'attribuer l'accès aux utilisateurs et aux groupes de votre instance d'organisation IAM Identity Center. Vous attribuez l'accès à l'aide des rôles IAM de votre Comptes AWS. Vous pouvez utiliser le gestionnaire d'accès au compte conjointement avec les ensembles d'autorisations IAM Identity Center ou seul. Les ensembles d'autorisations Account Access Manager et IAM Identity Center utilisent les utilisateurs et les groupes que vous avez synchronisés à partir de votre source d'identité ou créés directement dans IAM Identity Center.

Pourquoi utiliser le gestionnaire d'accès au compte ?

Le gestionnaire d'accès aux comptes vous permet d'attribuer vos rôles IAM existants aux utilisateurs et aux groupes que vous avez synchronisés dans IAM Identity Center à partir de votre source d'identité d'entreprise. L'utilisation de vos rôles IAM existants vous offre une flexibilité supplémentaire pour personnaliser l'accès des utilisateurs, notamment :

- Configurez les balises de rôle IAM et utilisez les balises de session pour le contrôle d'accès basé sur les attributs (ABAC).
- Configurez la politique de confiance des rôles IAM pour restreindre l'attribution des rôles ou utiliser les IdP-asserted revendications comme conditions de politique de confiance.
- Configurez les chemins de rôles IAM pour regrouper les rôles de manière logique et simplifier l'administration. Pour plus d'informations, voir Configurer les chemins de rôles IAM pour regrouper les rôles de manière logique et simplifier l'administration.

Le tableau suivant compare les trois options d'accès au compte :

	Rôles IAM + gestionnaire d'accès au compte	Ensembles d'autorisations IAM Identity Center	Rôles IAM + fédération IAM (hypothèse de rôle directe)
Idéal pour	Rôles IAM personnalisés qui varient selon les comptes	Accès humain de base uniforme, par exemple en lecture seule et provisionné par l'administrateur sous forme de rôles IAM immuables	Adapté aux charges de travail
Scope (Portée)	Comptes multiples (AWS organisation)	Comptes multiples (AWS organisation)	Par compte
Approvisionnement des rôles IAM	Nécessite une infrastructure sous forme de code (IaC) ou un provisionnement manuel via la console AWS IAM	Provisionné automatiquement	Nécessite une infrastructure sous forme de code (IaC) ou un provisionnement manuel via AWS IAM et des consoles IdP externes
Role-to-identity le mappage se trouve dans	Gestionnaire d'accès au compte	IAM Identity Center	IdP externe
L'expérience de l'utilisateur final	Les utilisateurs accèdent aux AWS comptes via l' AWS URL du portail d'accès aux comptes après s'être connectés en AWS tant qu'utilisateurs d'IAM Identity Center.	Les utilisateurs accèdent à leurs AWS comptes via le portail AWS d'accès après s'être connectés en AWS tant qu'utilisateurs d'IAM Identity Center.	Les utilisateurs accèdent à leurs AWS comptes via des applications SAML spécifiques au compte (icônes) sur leur portail IdP.

	Rôles IAM + gestionnaire d'accès au compte	Ensembles d'autorisations IAM Identity Center	Rôles IAM + fédération IAM (hypothèse de rôle directe)
AWS CLI expérience utilisateur	1. L'utilisateur se AWS connecte puis accède à un compte avec un rôle spécifique dans un navigateur Web. 2. L'utilisateur aws login saisit sur la ligne de commande pour récupérer les informations d'identification de session de rôle IAM à partir du navigateur. Pendant la durée de la session de rôle IAM, l'utilisateur peut continuer à utiliser le AWS CLI sans avoir à s'authentifier à nouveau à l'étape 1.	L'utilisateur configure un profil comprenant la paire de comptes et de rôles souhaitée en aws configure sso tapant. L'utilisateur aws sso login saisit pour lancer une nouvelle session à l'aide d'un profil. S'il n'y a pas de session IAM Identity Center active, l'utilisateur s'authentifie dans un navigateur et y autorise l'accès. Consultez le guide de AWS IAM Identity Center l'utilisateur pour plus de détails.	Intégrations personnalisées à l'aide de fournisseurs d'informations d' AWS identification SDK

Principales fonctionnalités

Portail d'accès au compte

Le gestionnaire d'accès aux comptes fournit aux utilisateurs un portail d'accès aux comptes où ils peuvent consulter et accéder aux informations qui leur Comptes AWS sont attribuées via le gestionnaire d'accès au compte.

Accès à la CLI

Votre personnel peut y accéder Comptes AWS à l' AWS CLI aide de la aws login commande après s'être AWS connecté dans un navigateur.

API d'accès au compte

Le gestionnaire d'accès aux comptes possède son propre espace de noms d'API`account-access`, qui est distinct de l'API IAM. Vous utilisez cet espace de noms dans les SDK AWS CLI et.

Commencer à utiliser le gestionnaire d'accès au compte

Ce qui suit explique comment démarrer avec le gestionnaire d'accès au compte. Cela suppose que les rôles IAM que vous souhaitez attribuer existent déjà dans leurs rôles respectifs Comptes AWS. Si ces rôles n'existent pas, voir Création d'un rôle à l'aide de politiques de confiance personnalisées pour plus d'informations sur la création de nouveaux rôles.

Pour commencer à utiliser le gestionnaire d'accès au compte

1. Assurez-vous qu'une instance d'organisation d'IAM Identity Center est activée et que les utilisateurs et les groupes de votre personnel y sont configurés. Consultez la section Premiers pas avec IAM Identity Center du Guide de l' AWS IAM Identity Center utilisateur pour plus d'informations. Pour plus de détails, voir plus the section called “Prérequis et considérations” loin dans cette rubrique.

   > **Note**
   >
   > Si vous utilisez IAM Identity Center répliqué dans plusieurs AWS régions ou avec une clé gérée par le client (CMK), vous devez `kms:Decrypt` autoriser le gestionnaire d'accès au compte à utiliser cette clé CMK lors de la connexion à IAM Identity Center. Pour plus

d'informations, consultez la politique de base relative aux clés KMS dans le Guide de AWS IAM Identity Center l'utilisateur.

2. Assurez-vous que vous disposez des autorisations nécessaires pour activer le gestionnaire d'accès aux comptes et gérer les attributions de comptes dans celui-ci. Pour de plus amples informations, veuillez consulter the section called "Gestion des identités et des accès pour le gestionnaire d'accès aux comptes".
3. Activez le gestionnaire d'accès au compte via la console ou l'API.
4. Mettez à jour vos pare-feux et passerelles pour accéder au portail d'accès au compte.
5. Autorisez le gestionnaire d'accès aux comptes à accéder aux rôles IAM en mettant à jour sa politique de confiance.
6. Commencez à attribuer l'accès aux comptes aux utilisateurs et aux groupes de votre personnel depuis IAM Identity Center.
7. Indiquez à vos utilisateurs comment accéder à Comptes AWS l'aide des attributions de rôles de gestionnaire d'accès aux comptes.

Prérequis et considérations

Les conditions préalables et considérations suivantes s'appliquent :

- Le gestionnaire d'accès aux comptes est disponible dans les régions AWS commerciales activées par défaut. Opt-in Les régions commerciales GovCloud (États-Unis), la Chine et les autres régions non commerciales ne sont pas prises en charge.
- Lors de l'activation, le gestionnaire d'accès au compte se connecte à une instance organisationnelle unique d'IAM Identity Center. Vous devez au préalable activer une AWS organisation et une instance d'organisation d'IAM Identity Center. Les instances de compte d'IAM Identity Center ne sont pas prises en charge.
- Nous vous recommandons d'activer le gestionnaire d'accès aux comptes dans la même AWS région que celle dans laquelle vous avez activé votre instance IAM Identity Center. Si ce n'est pas déjà fait, consultez la section Considérations relatives au choix d'une AWS région dans le Guide de AWS IAM Identity Center l'utilisateur.
- L'instance d'organisation d'IAM Identity Center et le gestionnaire d'accès au compte doivent être activés dans le compte de gestion de votre organisation.

Activer le gestionnaire d'accès au compte

Après avoir créé une instance d'organisation d'IAM Identity Center, vous pouvez activer le gestionnaire d'accès aux comptes via sa console ou son API. Vous devez effectuer cette opération dans la même Compte AWS région que votre instance IAM Identity Center.

Avant d'activer le gestionnaire d'accès au compte, exécutez la commande suivante depuis le compte de gestion de AWS l'organisation pour accorder l'accès au gestionnaire d'accès au compte à votre organisation :

```
aws organizations enable-aws-service-access --service-principal account-
access.amazonaws.com
```

> **Note**
>
> Bien que la console du gestionnaire d'accès aux comptes fasse partie de la console IAM, son API se trouve dans un espace de noms distinct.

Utilisez l'une des méthodes suivantes pour activer le gestionnaire d'accès au compte :

Console

Pour activer le gestionnaire d'accès au compte

1. Connectez-vous au compte de gestion de votre organisation.
2. Dans le volet de navigation de la Gestion des identités et des accès AWS console, choisissez Account access manager.
3. Vérifiez que la région de la console est définie sur la région principale de votre instance IAM Identity Center.
4. S'il n'existe aucune instance d'organisation d'IAM Identity Center, vous serez invité à en activer une dès maintenant avant de continuer. Sinon, choisissez Activer le gestionnaire d'accès au compte.

AWS CLI

Recherchez d'abord l'ARN de l'instance IAM Identity Center et vérifiez que le code de région correspond à la région principale d'IAM Identity Center. Ensuite, exécutez la commande suivante :

```
aws account-access create-application \
  --region <Region> \
  --identity-source '{
    "identityCenter": {
      "instanceArn": "<IAM_IDENTITY_CENTER_INSTANCE_ARN>"
    }
  }'
```

Mettre à jour les pare-feux et les passerelles

Le portail d'accès aux comptes fournit à vos utilisateurs un accès par authentification unique aux informations qui leur Comptes AWS sont attribuées via le gestionnaire d'accès au compte.

Si vous filtrez l'accès à des AWS domaines ou à des points de terminaison d'URL spécifiques à l'aide d'une solution de filtrage de contenu Web telle que des pare-feux de nouvelle génération (NGFW) ou des passerelles Web sécurisées (SWG), vous devez autoriser les domaines et les points de terminaison d'URL associés au portail d'accès au compte.

La liste suivante répertorie les domaines à double pile et les points de terminaison d'URL à ajouter à la liste d'autorisation de votre solution de filtrage de contenu Web. Le gestionnaire d'accès au compte ne possède aucun IPv4-only point de terminaison.

Dual-stack liste d'autorisations

URL du portail d'accès au compte :

- `https://[Tenant-ID].account-access.[Region].app.aws`

Les API que le portail d'accès aux comptes appelle :

- `[Tenant-ID].account-access.[Region].app.aws/api/*`
- `[Tenant-ID].account-access.[Region].app.aws/auth/*`

Tenant-ID est un identifiant aléatoire unique associé au gestionnaire d'accès au compte. Pour trouver sa valeur, accédez à l'onglet Paramètres de la console du gestionnaire d'accès aux comptes, puis recherchez l'URL de l'application dans la section Détails de l'instance. Tenant-ID est le premier sous-domaine de l'URL.

Les points de terminaison du service Account Access Manager :

- Control-plane: `account-access.[Region].api.aws`
- Runtime: `account-access-runtime.[Region].api.aws`
- Console : `console.aws.amazon.com/account-access`

Si vous utilisez le AWS CLI, vous pouvez trouver l'ID du locataire en exécutant la commande CLI suivante dans le compte de gestion ou d'administrateur délégué de votre organisation.

```
aws account-access list-applications
```

La réponse contient des informations sur le gestionnaire d'accès au compte, notamment l'ARN de l'application et l'ID du locataire.

```
{
    "applications": [
        {
            "applicationArn": "arn:aws:account-access:us-
west-2:123456789012:application/1234567890abcdef",
            "tenantId": "aa-gyxmap389",
            "createdAt": "2026-03-27T18:31:19+00:00",
            "updatedAt": "2026-03-27T18:31:19+00:00"
        }
    ]
}
```

Considérations relatives à l'autorisation de répertorier des domaines et des points de terminaison d'URL

Outre les exigences relatives à la liste d'autorisation pour le portail d'accès aux comptes, les autres services et applications que vous utilisez peuvent nécessiter la mise en liste des domaines autorisés.

- Vous devez répertorier le portail AWS d'accès et ses dépendances. Pour plus d'informations, consultez la section Mise à jour des pare-feux et des passerelles pour autoriser l'accès au portail AWS d'accès dans le Guide de l' AWS IAM Identity Center utilisateur.
- Pour accéder Comptes AWSà la Console de gestion AWS console du gestionnaire d'accès au compte et à la console, vous devez autoriser des domaines supplémentaires. Reportez-vous à la section Résolution des problèmes du Guide de démarrage pour Console de gestion AWS obtenir la liste des Console de gestion AWS domaines.

Supprimer le gestionnaire d'accès au compte

Lorsque le gestionnaire d'accès au compte est supprimé, toutes les données qu'il contient sont supprimées et ne peuvent pas être récupérées.

Suivez la procédure suivante pour supprimer le gestionnaire d'accès au compte.

Console

Pour supprimer l'instance du gestionnaire d'accès à votre compte

1. Connectez-vous au compte de gestion de votre organisation.
2. Dans le volet de navigation de la Gestion des identités et des accès AWS console, choisissez Account access manager.
3. Vérifiez que la région de la console est définie sur la région principale de votre instance de gestionnaire d'accès au compte.
4. Sélectionnez l'onglet Paramètres.
5. Dans la section Supprimer le gestionnaire d'accès au compte, choisissez Supprimer.
6. La boîte de dialogue vous invite à confirmer. Lorsque vous êtes prêt, choisissez Supprimer.

AWS CLI

Exécutez la commande suivante pour supprimer le gestionnaire d'accès au compte :

```
aws account-access delete-application \
  --region "<Region>" \
  --application-arn "<account_access_manager_ARN>"
```

⚠ Warning

La suppression du gestionnaire d'accès au compte est irréversible et a les conséquences suivantes :

- Toutes les attributions de rôles sont définitivement supprimées. Vos utilisateurs perdent immédiatement la possibilité d'accéder à l' Comptes AWS aide des rôles IAM attribués par le biais du gestionnaire d'accès au compte.

- Le gestionnaire d'accès au compte est supprimé du portail AWS d'accès pour tous les utilisateurs.
- Les sessions actives des rôles IAM attribués via le gestionnaire d'accès aux comptes se poursuivent jusqu'à leur expiration, mais les utilisateurs ne peuvent pas démarrer de nouvelles sessions.
- Si vous réactivez le gestionnaire d'accès au compte ultérieurement, vous devez recréer toutes les attributions à partir de zéro. Les affectations précédentes ne sont pas récupérables.

Avant de procéder à la suppression, pensez à désactiver l'accès des utilisateurs à des comptes individuels afin de restreindre temporairement l'accès.

Gérez l'accès à Comptes AWS

Le gestionnaire d'accès aux comptes est intégré à IAM Identity Center et AWS Organizations permet de gérer de manière centralisée l'attribution des rôles IAM existants sur plusieurs comptes Comptes AWS sans configurer chacun de vos comptes individuellement. Avec le gestionnaire d'accès aux comptes, vous attribuez des rôles IAM existants dans votre compte aux utilisateurs et Comptes AWS aux groupes IAM Identity Center afin de contrôler leur accès à des utilisateurs et à des groupes spécifiques. Comptes AWS

Grâce au gestionnaire d'accès aux comptes, vos équipes peuvent créer leurs propres rôles IAM personnalisés dans leur environnement et Comptes AWS, en tant qu'administrateur, vous utilisez le gestionnaire d'accès aux comptes pour gérer de manière centralisée les attributions des utilisateurs et des groupes IAM Identity Center à ces rôles IAM.

> **Note**
>
> Dans le contexte du gestionnaire d'accès aux comptes, les termes utilisateurs et groupes désignent exclusivement les utilisateurs et les groupes de personnel dans IAM Identity Center.

Administration déléguée

Lorsque vous activez le gestionnaire d'accès au compte, il le AWS crée dans le compte de gestion AWS Organizations car le gestionnaire d'accès au compte gère les attributions de rôles IAM pour AWS l'ensemble d'une organisation. Vous pouvez choisir de déléguer l'administration du gestionnaire d'accès au compte à un compte membre dans AWS Organizations.

Si vous choisissez d'enregistrer un compte membre en tant qu'administrateur délégué pour le gestionnaire d'accès au compte, les utilisateurs de ce compte peuvent effectuer la plupart des tâches administratives du gestionnaire d'accès au compte. Le tableau suivant indique si une tâche administrative du gestionnaire d'accès aux comptes peut être effectuée par les utilisateurs d'un compte de gestion d'organisation, d'un compte de membre d'administration déléguée ou d'un autre compte de membre. La gestion des rôles IAM est exclue de la liste des tâches administratives car elle s'effectue en dehors du gestionnaire d'accès au compte.

Tâches administratives du gestionnaire d'accès aux comptes	Compte de gestion	Compte administrateur délégué	Compte membre
Gérer les attributions de rôles IAM*	Oui	Oui	Non
Activer le gestionnaire d'accès au compte	Oui	Non	Non
Supprimer le gestionnaire d'accès au compte	Oui	Non	Non
Activer ou désactiver l'accès des utilisateurs dans le compte de gestion	Oui	Non	Non
Activer ou désactiver l'accès utilisateur à un compte membre	Oui	Oui	Non
Enregistrer ou annuler l'enregistrement d'un compte de membre	Oui	Non	Non

Tâches administratives du gestionnaire d'accès aux comptes	Compte de gestion	Compte administrateur délégué	Compte membre
en tant qu'administrateur délégué			

*Reportez-vous aux meilleures pratiques en matière d'administration déléguée concernant les attributions d'utilisateurs et de groupes au compte de gestion.

Enregistrer un compte de membre

Pour configurer l'administration déléguée, vous devez d'abord enregistrer un compte membre dans votre organisation en tant qu'administrateur délégué. Les utilisateurs de ce compte membre qui disposent d'autorisations suffisantes auront un accès administratif au gestionnaire d'accès au compte.

Le gestionnaire d'accès aux comptes prend en charge l'enregistrement d'un seul compte membre en tant qu'administrateur délégué à la fois. Vous ne pouvez créer un compte membre que si vous êtes connecté avec les informations d'identification du compte de gestion.

Suivez la procédure suivante pour enregistrer un compte de membre dans votre AWS organisation en tant qu'administrateur délégué.

Pour créer un compte de membre

1. Connectez-vous à l' Console de gestion AWS aide des informations d'identification de votre compte de gestion AWS Organizations. Les informations d'identification du compte de gestion sont requises pour exécuter l'RegisterDelegatedAdministratorAPI.
2. Sélectionnez la région dans laquelle le gestionnaire d'accès au compte a été activé, puis ouvrez la Gestion des identités et des accès AWS console.
3. Choisissez Gestionnaire d'accès au compte, puis choisissez l'onglet Paramètres.
4. Dans la section Administrateur délégué, choisissez Enregistrer.
5. Sur la page Enregistrer l'administrateur délégué, sélectionnez celui que Compte AWS vous souhaitez enregistrer, puis choisissez Enregistrer.

Annuler l'enregistrement d'un compte membre

Vous ne pouvez annuler l'enregistrement d'un compte de membre que lorsque vous êtes connecté avec les informations d'identification du compte de gestion.

Suivez la procédure suivante pour annuler l'enregistrement d'un compte de membre de votre AWS organisation qui avait été précédemment désigné en tant qu'administrateur délégué.

> **Important**
>
> Lorsque vous désenregistrez un compte, vous supprimez la possibilité pour tous les utilisateurs administrateurs de gérer le gestionnaire d'accès au compte depuis ce compte. Ils ne peuvent plus administrer Compte AWS l'accès depuis ce compte.
> Cette opération n'affecte pas les autorisations ou les attributions configurées dans le gestionnaire d'accès au compte. Vos utilisateurs finaux continuent d'accéder à leur compte Comptes AWS depuis le portail d'accès au compte.

Pour annuler l'enregistrement d'un compte de membre

1. Connectez-vous à l' Console de gestion AWS aide des informations d'identification de votre compte de gestion AWS Organizations. Les informations d'identification du compte de gestion sont requises pour exécuter l'DeregisterDelegatedAdministratorAPI.
2. Sélectionnez la région dans laquelle le gestionnaire d'accès au compte a été activé, puis ouvrez la Gestion des identités et des accès AWS console.
3. Choisissez Gestionnaire d'accès au compte, puis choisissez l'onglet Paramètres.
4. Dans la section Administrateur délégué, choisissez Désenregistrer le compte.
5. Dans la boîte de dialogue Désenregistrer l'administrateur délégué, examinez les implications en matière de sécurité, puis confirmez que vous avez bien compris.
6. Choisissez Deregister (Annuler l'enregistrement).

Afficher le compte d'administrateur délégué enregistré

Suivez la procédure suivante pour déterminer quel compte de membre de votre compte AWS Organizations a été configuré en tant qu'administrateur délégué pour le gestionnaire d'accès au compte.

Pour consulter votre compte de membre enregistré

1. Connectez-vous à l' Console de gestion AWS aide des informations d'identification de votre compte de gestion AWS Organizations.
2. Sélectionnez la région dans laquelle le gestionnaire d'accès au compte a été activé, puis ouvrez la Gestion des identités et des accès AWS console.
3. Choisissez Gestionnaire d'accès au compte, puis choisissez l'onglet Paramètres.
4. Dans la section Administrateur délégué, vous pouvez consulter les détails du compte administrateur délégué actuel.

Préparez vos rôles IAM

Avant de commencer à attribuer des rôles IAM aux utilisateurs et aux groupes IAM Identity Center, vous devez :

1. Identifier les rôles IAM pertinents dans les comptes d'intérêt, et
2. Assurez-vous que la politique de confiance des rôles IAM contient la déclaration ci-dessous.

Si le rôle possède déjà une politique de confiance, ajoutez cette instruction à son tableau Statement. Si vous créez un nouveau rôle, vous pouvez utiliser l'intégralité du document de politique. Cette déclaration de politique permet au gestionnaire d'accès au compte d'assumer le rôle et de définir le contexte pour le compte des utilisateurs. Le `Condition` bloc fournit une protection confuse en tant qu'adjoint en garantissant que le gestionnaire d'accès au compte ne peut assumer le rôle que pour le compte de votre propre compte et de votre application de gestion d'accès au compte. Pour plus d'informations, voir Mettre à jour une politique de confiance relative aux rôles.

Remplacez-le `<your-account-id>` par votre Compte AWS identifiant, `<region>` par votre région principale IAM Identity Center et `<application-id>` par l'ID de votre application de gestion d'accès au compte. L'ARN de l'application se trouve sur la page Paramètres de la console du gestionnaire d'accès au compte.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountAccessManagerIAMRoleTrustPolicyStatement",
      "Effect": "Allow",
```

```
      "Principal": {
        "Service": "account-access.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetContext"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<your-account-id>",
          "aws:SourceArn": "arn:aws:account-access:<region>:<your-account-
id>:application/<application-id>"
        }
      }
    }
  ]
}
```

Facultatif — **`sts:TagSession`**. Ajoutez `sts:TagSession` à la `Action` liste si vous souhaitez que les attributs d'assertion en amont de votre fournisseur d'identité (par exemple, les adhésions à un groupe, un service, un centre de coûts) soient propagés en tant que balises principales sur la session émise par le gestionnaire d'accès au compte. Sans cette action, l'hypothèse du rôle fonctionne toujours ; seule la propagation des balises de session est supprimée. Ajoutez-le lorsque les politiques IAM en aval du compte cible utilisent des balises principales (`aws:PrincipalTag/<key>`) ; sinon, omettez-le. Gardez le même `Condition` bloc.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountAccessManagerIAMRoleTrustPolicyStatement",
      "Effect": "Allow",
      "Principal": {
        "Service": "account-access.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetContext",
        "sts:TagSession"
      ],
      "Condition": {
        "StringEquals": {
```

```
          "aws:SourceAccount": "<your-account-id>",
          "aws:SourceArn": "arn:aws:account-access:<region>:<your-account-
id>:application/<application-id>"
        }
      }
    }
  ]
}
```

> **Note**
>
> Nous vous recommandons de mettre à jour vos flux de création de rôles IAM pour ajouter automatiquement cette déclaration de politique de confiance si nécessaire.

Attribuer et supprimer l'accès

Attribuer l'accès d'un utilisateur ou d'un groupe à un Compte AWS

Vous pouvez attribuer un accès à un utilisateur ou à un groupe Compte AWS en créant une attribution entre l'utilisateur ou le groupe, le compte et le rôle IAM que l'utilisateur ou le groupe peut assumer dans le compte. Ainsi, une seule attribution est un triplet composé de l'ID d'utilisateur ou de groupe, de l'ID de compte et de l'ARN du rôle IAM.

Si vous souhaitez effectuer des attributions groupées, par exemple en attribuant le même rôle à plusieurs utilisateurs et groupes, vous devez créer chaque attribution individuellement. Vous pouvez créer un script pour cette procédure à l'aide de l'API d'accès au compte et du AWS SDK/CLI.

Console

La console propose différentes manières d'attribuer un nouvel accès, selon que vous souhaitez d'abord identifier le compte, l'utilisateur ou le groupe.

> **Note**
>
> La console du gestionnaire d'accès aux comptes fournit une fonction de recherche pour les utilisateurs, les groupes et les comptes. Pour identifier le rôle que vous souhaitez attribuer, vous devez trouver son nom (ou son ARN) au préalable. La console du gestionnaire d'accès au compte ne peut pas détecter les rôles IAM dans Comptes AWS.

Pour toutes les procédures de cette section, connectez-vous au compte de gestion (ou au compte administrateur délégué), sélectionnez la région dans laquelle le gestionnaire d'accès au compte est activé et accédez au gestionnaire d'accès au compte dans la console IAM.

Accounts tab

Pour attribuer un accès à un utilisateur ou à un groupe depuis l'onglet Comptes

1. Sélectionnez l'onglet Accounts.
2. Si vous connaissez déjà l'identifiant du compte, choisissez Attribuer un nouvel accès. Sur la page Attribuer l'accès au compte, recherchez l'utilisateur ou le groupe souhaité, entrez l'ID du compte et le nom du rôle IAM, puis choisissez Attribuer l'accès.
3. Si vous devez d'abord trouver le compte, vous pouvez rechercher le compte souhaité Compte AWS par son nom, son identifiant ou son adresse e-mail. Vous pouvez également consulter la structure organisationnelle ou la liste des comptes pour trouver ceux qui vous conviennent Compte AWS. Cliquez sur le bouton radio situé devant le compte, puis sur Attribuer un nouvel accès. Sur la page Attribuer l'accès au compte, recherchez l'utilisateur ou le groupe souhaité, entrez le nom du rôle IAM, puis choisissez Attribuer l'accès.
4. La structure organisationnelle ou la liste des comptes vous permet de choisir un compte pour afficher les détails du compte et les attributions d'accès au compte. Vous pouvez également choisir Attribuer un nouvel accès à partir de là.

Users tab

Pour attribuer l'accès aux utilisateurs depuis l'onglet Utilisateurs

1. Sélectionnez l’onglet Utilisateurs.
2. Vous pouvez rechercher des utilisateurs par nom d'utilisateur et nom d'affichage ou les trouver dans la liste des utilisateurs affichée.
3. Choisissez l'utilisateur auquel vous souhaitez attribuer un nouvel accès.
4. La page suivante présente les attributs des utilisateurs, les attributions de comptes et les appartenances aux groupes.
5. Choisissez Attribuer un nouvel accès.
6. Sur la page Attribuer l'accès au compte, entrez l'ID du compte et le nom du rôle IAM, puis choisissez Attribuer l'accès.

Groups tab

Pour attribuer l'accès à un groupe depuis l'onglet Groupes

1. Dans le volet de navigation, choisissez Account access manager.
2. Cliquez sur l'onglet Groups (Groupes).
3. Vous pouvez rechercher des groupes par nom de groupe ou les trouver dans la liste des groupes affichée.
4. Choisissez le groupe auquel vous souhaitez attribuer un nouvel accès.
5. La page suivante présente les attributs des groupes, les attributions de comptes et les membres des utilisateurs.
6. Choisissez Attribuer un nouvel accès.
7. Sur la page Attribuer l'accès au compte, entrez l'ID du compte et le nom du rôle IAM, puis choisissez Attribuer l'accès.

AWS CLI

La AWS CLI commande suivante peut être utilisée pour créer une attribution de compte pour un utilisateur :

```
aws account-access create-entitlement \
  --region <Region> \
  --application-arn "<Account_access_manager_ARN>" \
  --entitlement '{
    "principalRole": {
      "principal": {
        "identityCenter": {
          "userId": "<Identity_Center_User_ID>"
        }
      },
      "roleArn": "arn:aws:iam::<ACCOUNT_ID>:role/<ROLE_NAME>"
    }
  }'
```

La AWS CLI commande suivante peut être utilisée pour créer une attribution de compte pour un groupe :

```
aws account-access create-entitlement \
  --region <Region> \
```

```
--application-arn "<Account_access_manager_ARN>" \
--entitlement '{
  "principalRole": {
    "principal": {
      "identityCenter": {
        "groupId": "<IDC_GROUP_ID>"
      }
    },
    "roleArn": "arn:aws:iam::<ACCOUNT_ID>:role/<ROLE_NAME>"
  }
}'
```

Supprimer l'accès d'un utilisateur ou d'un groupe à un Compte AWS

Vous pouvez trouver l'affectation à supprimer de plusieurs manières.

Console

Pour toutes les procédures de cette section, connectez-vous au compte de gestion (ou au compte administrateur délégué), sélectionnez la région dans laquelle le gestionnaire d'accès au compte est activé et accédez au gestionnaire d'accès au compte dans la console IAM.

Accounts tab

Pour supprimer l'accès d'un utilisateur ou d'un groupe depuis l'onglet Comptes

1. Sélectionnez l'onglet Accounts.
2. Trouvez le compte concerné et choisissez son nom.
3. Dans la section Attributions d'accès au compte, choisissez l'utilisateur ou le groupe pour lequel vous souhaitez supprimer l'accès, puis choisissez Annuler l'attribution.
4. Dans le modal, choisissez Annuler l'attribution pour confirmer.

Users tab

Pour supprimer l'accès des utilisateurs depuis l'onglet Utilisateurs

1. Sélectionnez l'onglet Utilisateurs.
2. Choisissez l'utilisateur pour lequel vous souhaitez supprimer l'accès.
3. Dans la section Attributions d'accès au compte, choisissez le compte pour lequel vous souhaitez supprimer l'accès, puis choisissez Annuler l'attribution.

4. Dans le modal, choisissez Annuler l'attribution pour confirmer.

AWS CLI

Pour supprimer l'accès à l'aide de la CLI, vous devez d'abord rechercher l'attribution du compte (droit), puis utiliser son identifiant dans la commande suivante. Vous pouvez voir plus loin dans cette rubrique comment rechercher des attributions de compte pour un compte, un utilisateur, un groupe ou un rôle spécifique. La CLI fait référence aux affectations en tant que droits.

```
aws account-access delete-entitlement \
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --entitlement-id "<ENTITLEMENT_ID>"
```

Afficher ou rechercher des attributions de comptes

Vous pouvez consulter les attributions de comptes de différentes manières.

Console

Pour toutes les procédures de cette section, connectez-vous au compte de gestion (ou au compte administrateur délégué), sélectionnez la région dans laquelle le gestionnaire d'accès au compte est activé et accédez au gestionnaire d'accès au compte dans la console IAM.

Accounts tab

Afficher l'accès des utilisateurs ou des groupes depuis l'onglet Comptes

1. Sélectionnez l'onglet Accounts.
2. Par défaut, la vue Hiérarchie s'affiche. Vous pouvez passer à l'affichage en liste si vous le souhaitez. Vous pouvez rechercher des comptes par nom, identifiant, e-mail et identifiant d'unité d'organisation. Lorsque vous trouvez le compte souhaité, choisissez son nom.
3. Sur la page du compte, vous pouvez voir les attributions de compte dans la section Attributions d'accès au compte. Vous pouvez les filtrer davantage par nom d'utilisateur, ID utilisateur, nom de groupe, ID de groupe et ARN du rôle IAM.

Users tab

Afficher l'accès des utilisateurs depuis l'onglet Utilisateurs

1. Sélectionnez l’onglet Utilisateurs.
2. Recherchez l'utilisateur par son nom ou son identifiant. Lorsque vous trouvez l'utilisateur souhaité, choisissez son nom d'utilisateur.
3. Sur la page utilisateur, vous pouvez voir les attributions de compte de l'utilisateur. Vous pouvez les filtrer davantage en fonction de l'ID de compte, du nom du compte et de l'ARN du rôle IAM.

Groups tab

Afficher l'accès aux groupes depuis l'onglet Groupes

1. Cliquez sur l'onglet Groups (Groupes).
2. Recherchez le groupe par nom ou par identifiant de groupe. Lorsque vous avez trouvé le groupe souhaité, choisissez son nom.
3. Sur la page du groupe, vous pouvez voir les attributions des comptes du groupe. Vous pouvez les filtrer davantage en fonction de l'ID de compte, du nom du compte et de l'ARN du rôle IAM.

AWS CLI

Les commandes suivantes montrent comment récupérer les attributions de comptes à l'aide de différents filtres. La CLI fait référence aux affectations en tant que droits.

Récupérez les 100 premières affectations (droits) pour un compte spécifique

```
aws account-access list-entitlements \
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --filter '{
      "principalRole": {
        "account": "<ACCOUNT_ID>"
      }
    }' \
    --max-results 100
```

Récupérez toutes les attributions (droits) pour un utilisateur spécifique

```
aws account-access list-entitlements \
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --filter '{
      "principal": {
        "type": "USER",
        "id": "<USER_ID>"
      }
    }'
```

Récupérez tous les devoirs (droits) pour un groupe spécifique

```
aws account-access list-entitlements \
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --filter '{
      "principal": {
        "type": "GROUP",
        "id": "<GROUP_ID>"
      }
    }'
```

Sessions et contrôle d'accès

Séances de compréhension

Les utilisateurs qui accèdent à Comptes AWS l'aide des attributions du gestionnaire d'accès aux comptes lancent les types de session suivants :

1. Session de rôle IAM : le rôle IAM assumé par un utilisateur est un identifiant temporaire pour accéder aux AWS ressources, d'une durée par défaut d'une heure.
2. Session du portail d'accès au compte — Les utilisateurs initient cette session lorsqu'ils utilisent le portail d'accès au compte. Il est géré en tant que session d'application par IAM Identity Center.
3. AWS session du portail d'accès (également appelée session interactive utilisateur) : cette session est gérée par IAM Identity Center.
4. Session de source d'identité externe (IdP ou Microsoft AD) : cette session est applicable lorsque l'instance IAM Identity Center est connectée à une source d'identité externe à des fins d'authentification.

Pour plus d'informations sur les sessions interactives et applicatives avec les utilisateurs dans IAM Identity Center, y compris la durée des sessions, consultez la section Présentation des sessions d'authentification dans IAM Identity Center dans le Guide de l'AWS IAM Identity Center utilisateur.

> **Important**
>
> Les sessions des rôles IAM attribués par le gestionnaire d'accès aux comptes fonctionnent sur la base de la durée de session par défaut d'une heure, indépendamment de la durée de session maximale définie dans le rôle IAM, de l'état des sessions dans le portail d'accès au compte et le portail d' AWS accès.

Révoquer l'accès utilisateur

Pour révoquer l'accès utilisateur attribué par le biais des attributions de rôles de gestionnaire d'accès au compte, voir Révoquer l'accès utilisateur dans le Guide de l'AWS IAM Identity Center utilisateur.

Attribute-based contrôle d'accès

Vous pouvez configurer votre fournisseur d'identité (IdP) pour qu'il envoie des attributs sous forme de balises de session via des assertions SAML en définissant le nom de l'attribut sur`https://aws.amazon.com/SAML/Attributes/AccessControl:TagKey`, où se *`TagKey`* trouve la clé de balise de session que vous souhaitez renseigner. IAM Identity Center transmet le nom et la valeur de l'attribut de l'IdP à des fins d'évaluation des politiques.

Accès à la main-d'œuvre

Le gestionnaire d'accès aux comptes vous permet d'attribuer Compte AWS l'accès aux utilisateurs et aux groupes de votre personnel dans IAM Identity Center en les mappant aux rôles IAM de votre. Comptes AWS Si vous utilisez IAM Identity Center pour la première fois, consultez Configuration de l'accès du personnel aux AWS ressources dans le Guide de l'AWS IAM Identity Center utilisateur pour obtenir des informations sur les sessions d'authentification, la gestion de l'accès des utilisateurs, la gestion des mots de passe et l'authentification multifacteur.

Portails utilisateurs

Pour accéder aux AWS comptes qui leur sont attribués, les utilisateurs de votre personnel peuvent accéder directement au portail d'accès aux comptes à l'aide d'un signet, ou le lancer depuis le portail AWS d'accès.

Vos utilisateurs peuvent interagir avec deux portails utilisateur :

Portail d'accès au compte

Il s'agit d'un portail (URL) qui permet aux utilisateurs de la main-d'œuvre d'accéder aux informations qui leur Comptes AWS sont attribuées à l'aide du gestionnaire d'accès au compte. Ce portail (URL) peut être ajouté aux favoris pour un accès direct et est également disponible dans le portail AWS d'accès en tant qu'application AWS gérée.

AWS portail d'accès

Il s'agit du principal point d'entrée AWS pour les utilisateurs d'IAM Identity Center. Ce portail affiche toutes les AWS applications auxquelles les utilisateurs ont accès, y compris le portail d'accès au compte.

> **Tip**
>
> Nous vous recommandons d'ajouter les deux portails à vos favoris pour une navigation plus rapide. Si vous utilisez un fournisseur d'identité externe, vous pouvez également y créer des applications de favoris pour les deux portails.

Commencer à utiliser le portail d'accès au compte

Trouver l'URL du portail d'accès au compte

Pour trouver l'URL du portail d'accès au compte, choisissez l'onglet Paramètres dans la console du gestionnaire d'accès au compte. Recherchez l'URL de l'application dans la section Détails de l'instance. L'URL suit le modèle`https://[Tenant-ID].account-access.[Region].app.aws`. Tenant-ID est un identifiant aléatoire unique associé au gestionnaire d'accès au compte.

Si vous utilisez le AWS CLI, vous pouvez trouver l'ID du locataire en exécutant la commande CLI suivante dans le compte de gestion ou d'administrateur délégué de votre organisation.

```
aws account-access list-applications
```

La réponse contient des informations sur le gestionnaire d'accès au compte, notamment l'ARN de l'application et l'ID du locataire. Utilisez l'ID du locataire et le code régional du gestionnaire d'accès au compte pour créer l'URL de l'application. Suivez le schéma décrit plus haut dans cette section.

```
{
    "applications": [
        {
            "applicationArn": "arn:aws:account-access:us-
west-2:123456789012:application/1234567890abcdef",
            "tenantId": "aa-gyxmap389",
            "createdAt": "2026-03-27T18:31:19+00:00",
            "updatedAt": "2026-03-27T18:31:19+00:00"
        }
    ]
}
```

Configuration de l'URL d'un portail personnalisé

Pour obtenir des instructions sur la mise en œuvre de domaines personnalisés, consultez la section Routage régional pour les portails AWS d'accès : Implémentation de domaines personnalisés pour IAM Identity Center sur le blog AWS de sécurité. Vous pouvez adapter ces instructions au modèle d'URL du portail d'accès au compte.

Communiquer avec vos utilisateurs

Communiquez les informations suivantes aux utilisateurs de votre personnel :

- URL du portail d'accès au compte.
- Le portail d'accès au compte est également disponible pour vos utilisateurs sous forme d'application dans l'onglet Applications du portail AWS d'accès.
- Si vous avez activé le lancement du portail d'accès au compte à partir d'un portail de fournisseur d'identité externe ou d'un autre portail externe AWS, incluez ces informations dans la communication.

Accès utilisateur à Comptes AWS

Cette rubrique décrit comment les utilisateurs accèdent aux informations qui leur Comptes AWS sont attribuées à l'aide du gestionnaire d'accès au compte.

Obtenir l'URL de votre portail

Vous pouvez accéder à vos tâches Comptes AWS de plusieurs manières :

1. Votre administrateur peut vous fournir l'URL d'un portail d'accès au compte (par exemple, `https://aa-gyxmap389.account-access.us-east-2.app.aws` ou une URL personnalisée).
2. Votre administrateur peut vous fournir l'URL du portail d' AWS accès principal qui affiche toutes les AWS applications qui vous sont attribuées, y compris le portail d'accès au compte.
3. Votre organisation vous a peut-être autorisé à lancer le portail d'accès aux comptes depuis le portail d'un fournisseur d'identité externe ou un autre portail externe AWS.

Accès Comptes AWS à l'aide d'un navigateur Web

Le portail d'accès aux comptes affiche les informations auxquelles Comptes AWS vous avez accès et les rôles IAM que vous pouvez assumer pour chaque compte.

Pour accéder à un compte dans un navigateur

1. Accédez à l'URL du portail d'accès au compte et, lorsque vous y êtes invité, connectez-vous à l'aide de vos informations d'identification conformément aux instructions de votre administrateur.

   > **Note**
   >
   > Pour lancer le portail d'accès au compte à partir du portail AWS d'accès, accédez plutôt à l'onglet Applications.

2. La page du navigateur affiche les Comptes AWS informations qui vous sont attribuées. Vous pouvez rechercher des comptes par nom de compte ou par identifiant.
3. Développez le compte qui vous intéresse pour voir tous les rôles que vous pouvez y assumer.
4. Choisissez le rôle IAM avec lequel vous souhaitez vous connecter (par exemple, développeur). Un nouvel onglet de navigateur s'ouvre dans lequel vous êtes connecté au compte avec le rôle choisi.

Accès Comptes AWS avec le AWS CLI or AWS Kits SDK

Vous pouvez accéder aux AWS services par programmation à l'aide des kits de développement logiciel (SDK) AWS Command Line Interface ou des kits de développement AWS logiciel (SDK) avec les informations d'identification utilisateur d'IAM Identity Center.

Le portail d'accès aux comptes fournit aux utilisateurs un accès par authentification unique à leurs quatre attributions Comptes AWS de comptes gérées dans le gestionnaire d'accès aux comptes. Une fois connecté au portail d'accès au compte, vous pouvez obtenir des informations d'identification temporaires. Vous pouvez ensuite utiliser les informations d'identification contenues dans le AWS CLI ou AWS les kits SDK pour accéder aux ressources d'un Compte AWS.

Si vous utilisez le AWS CLI pour accéder aux AWS services par programmation, vous pouvez utiliser les procédures décrites dans cette rubrique pour lancer l'accès au. AWS CLI Pour plus d'informations à ce sujet AWS CLI, consultez le Guide de AWS Command Line Interface l'utilisateur.

Si vous utilisez les AWS kits de développement logiciel pour accéder à des AWS services par programmation, le fait de suivre les procédures décrites dans cette rubrique permet également d'établir directement l'authentification pour les kits de développement logiciel. AWS Pour plus d'informations sur les AWS kits de développement logiciel, consultez le Guide de référence AWS des kits de développement logiciel et des outils.

Compte AWS accès avec connexion AWS (recommandé)

Vous pouvez utiliser la aws login commande pour accéder à l'accès qui Comptes AWS vous est attribué dans le gestionnaire d'accès au compte. Tout d'abord, installez la dernière version AWS CLI. Suivez ensuite cette procédure :

Pour accéder à un compte avec AWS CLI

1. Dans votre navigateur par défaut, connectez-vous au portail d'accès à votre compte, puis connectez-vous à la cible Compte AWS en utilisant l'un des rôles IAM attribués au compte. Suivez la procédure décrite plus haut dans cette rubrique.
2. Sur la ligne de commande, exécutez la aws login commande.
3. Si vous n'avez pas défini de région par défaut, vous êtes AWS CLI invité à spécifier la AWS région de votre choix (par exemple, us-east-2). Ils AWS CLI mémoriseront votre choix pour une utilisation future.
4. Une page Web AWS CLI s'ouvre dans votre navigateur par défaut et vous invite à poursuivre la session active que vous venez de démarrer ou à vous connecter à une nouvelle session.
5. Sur la page Web, choisissez la vignette correspondant à la session que vous avez initiée plus tôt dans le navigateur.
6. La page du navigateur confirme que vos informations d'identification ont été partagées avec succès et peuvent être utilisées jusqu'à l'expiration de votre session. Vous pouvez maintenant fermer l'onglet contenant la page Web et continuer à AWS CLI utiliser cette session.

> **Note**
>
> Les Center-specific AWS CLI commandes IAM Identity telles que aws sso configure et aws sso login ne fonctionnent pas avec les attributions de comptes dans le gestionnaire d'accès aux comptes.

Compte AWS accès avec clés d'accès

Vous pouvez utiliser la méthode d'actualisation manuelle des informations d'identification pour obtenir des informations d'identification temporaires pour un rôle IAM qui vous est attribué dans un domaine spécifique. Compte AWS Pour ce faire, copiez et collez les commandes requises pour les informations d'identification temporaires. Avec cette méthode, vous devez actualiser les informations d'identification temporaires manuellement.

Vous pouvez exécuter des AWS CLI commandes jusqu'à l'expiration de vos informations d'identification temporaires.

Pour obtenir des informations d'identification que vous actualisez manuellement

1. Connectez-vous au portail d'accès au compte comme expliqué dans Accès à l' Comptes AWS aide d'un navigateur Web.
2. Localisez celui Compte AWS à partir duquel vous souhaitez récupérer les informations d'accès et développez-le pour afficher le nom du rôle IAM (par exemple Developer).
3. Choisissez Générer une clé d'accès à côté du rôle souhaité.

   > **Note**
   >
   > Si aucune Compte AWS affectation ne s'affiche, il est probable qu'aucun compte ne vous ait encore été attribué. Dans ce cas, contactez votre administrateur et demandez-lui de vous attribuer l'accès.

4. Dans la boîte de dialogue Obtenir les informations d'identification, choisissez l'PowerShellonglet macOS et Linux , Windows ou en fonction du système d'exploitation sur lequel vous avez installé le AWS CLI.
5. Choisissez l'une des options suivantes :

   - Option 1 : définir les variables d' AWS environnement

Choisissez cette option pour remplacer tous les paramètres d'identification, y compris ceux des fichiers d'identification et des fichiers de configuration. Pour plus d'informations, consultez la section Variables d'environnement à configurer AWS CLI dans le Guide de AWS Command Line Interface l'utilisateur.

Pour utiliser cette option, copiez les commandes dans votre presse-papiers, collez-les dans la fenêtre de votre AWS CLI terminal, puis appuyez sur Entrée pour définir les variables d'environnement requises.

- Option 2 : utiliser des valeurs individuelles dans votre client AWS de service

  Choisissez cette option pour accéder aux AWS ressources depuis un client AWS de service. Pour plus d'informations, consultez la section Outils sur lesquels s'appuyer AWS.

  Pour utiliser cette option, copiez les valeurs dans votre presse-papiers, collez-les dans votre code et affectez-les aux variables appropriées pour votre SDK. Pour plus d'informations, consultez la documentation de votre API SDK spécifique.

Création de liens de raccourci vers AWS Destinations de la console de gestion

Le portail d'accès au compte offre la possibilité de créer un lien profond partageable.

Pour créer un lien de raccourci

1. Lorsque vous êtes connecté au portail d'accès au compte, cliquez sur le bouton Créer un lien de raccourci.
2. Dans la boîte de dialogue :

   a. Choisissez un Compte AWS en utilisant l'identifiant ou le nom du compte. Au fur et à mesure que vous tapez, un menu déroulant affiche les identifiants de compte et les noms correspondants auxquels vous pouvez accéder. Vous ne pouvez choisir qu'un compte auquel vous avez accès.

   b. (Facultatif) Choisissez un rôle IAM dans la liste déroulante. Il s'agit des rôles IAM qui vous ont été attribués pour le compte sélectionné. Si vous omettez de choisir le rôle, les utilisateurs sont invités à en sélectionner un qui leur est attribué pour le compte choisi à l'aide du lien de raccourci.

> **Note**
>
> Vous ne pouvez pas accorder un nouvel accès à l'aide de liens de raccourci. Les liens de raccourci fonctionnent uniquement avec les comptes et les rôles déjà attribués à l'utilisateur.

Construire en toute sécurité AWS Liens de raccourci vers la console de gestion avec encodage d'URL

Toutes les valeurs des paramètres de l'URL, y compris l'ID de compte, le nom du rôle IAM et l'URL de destination, doivent être URL-encoded égales à.

Les liens de raccourci étendent l'URL du portail d'accès au compte avec le chemin suivant :

```
/#/console?account_id=[account_ID]&role_name=[role_name]&destination=[destination_URL]
```

L'URL complète d'une AWS région commerciale suit le modèle suivant :

```
https://[Tenant-ID].account-access.[Region].app.aws/#/console?
account_id=[account_ID]&role_name=[role_name]&destination=[destination_URL]
```

Voici un exemple de lien de raccourci qui connecte un utilisateur au compte 123456789012 avec le nom du rôle S3FullAccess IAM et l'amène à la page d'accueil de la console S3 :

```
https://aa-gyxmap389.account-access.us-east-2.app.aws/#/console?
account_id=123456789012&role_name=S3FullAccess&destination=https%3A%2F
%2Fconsole.aws.amazon.com%2Fs3%2Fhome
```

Considérations

Les considérations suivantes s'appliquent à l'utilisation de liens de raccourci :

- Si vous partagez des liens de raccourci avec des collègues, ceux-ci doivent avoir les mêmes attributions de compte pour pouvoir utiliser les liens de la même manière.
- Les liens de raccourci sont des liens régionaux. Si la région référencée connaît une interruption, le lien risque de ne pas fonctionner.

- Les liens de raccourci contiennent l'identifiant du compte et le nom du rôle en texte brut. Déterminez si les politiques de votre organisation limitent l'emplacement de stockage de ces métadonnées.

Sécurité dans le gestionnaire d'accès au compte

Cette section fournit des informations de sécurité spécifiques au gestionnaire d'accès au compte. Il complète les informations contenues dansSécurité, qui couvre les sujets de sécurité applicables à l'IAM dans son ensemble.

Gestion des identités et des accès pour le gestionnaire d'accès aux comptes

Les sections suivantes décrivent les autorisations dont vous avez besoin pour :

- Administrez le gestionnaire d'accès au compte dans le compte de gestion de votre organisation
- Administrer le gestionnaire d'accès au compte à partir d'un compte administrateur délégué

Administration du gestionnaire d'accès au compte depuis le compte de gestion de votre organisation

Pour gérer les paramètres du gestionnaire d'accès au compte dans le compte de gestion de AWS l'organisation, vous devez disposer des autorisations suivantes :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AAMCreateApplication",
      "Effect": "Allow",
      "Action": "account-access:CreateApplication",
      "Resource": "*"
    },
    {
      "Sid": "AAMApplicationAndEntitlementManagement",
      "Effect": "Allow",
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteApplication",
```

```
        "account-access:DeleteEntitlement",
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements",
        "account-access:ListTagsForResource",
        "account-access:TagResource",
        "account-access:UntagResource"
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
      "Action": "account-access:ListApplications",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": [
        "sso:CreateApplication",
        "sso:DeleteApplication",
        "sso:DescribeApplication",
        "sso:DescribeInstance",
        "sso:DescribeRegion",
        "sso:ListInstances",
        "sso:PutApplicationAccessScope",
        "sso:PutApplicationAssignmentConfiguration",
        "sso:PutApplicationAuthenticationMethod",
        "sso:PutApplicationGrant"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityStoreActions",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
        "sso-directory:DescribeGroup",
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
```

```
        "sso-directory:ListGroupsForUser",
        "sso-directory:ListMembersInGroup",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentKMSActions",
      "Effect": "Allow",
      "Action": "kms:Decrypt",
      "Resource": "*"
    },
    {
      "Sid": "CreateServiceLinkedRole",
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "arn:aws:iam::111122223333:role/aws-service-role/account-
access.amazonaws.com/AWSServiceRoleForAccountAccessManager",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "account-access.amazonaws.com"
        }
      }
    }
  ]
}
```

Facultatif : pour contraindre `account-access:CreateApplication` par balise, remplacez l' AAMCreateApplicationinstruction par :

```
{
  "Sid": "AAMCreateApplicationConstrainedByTag",
  "Effect": "Allow",
  "Action": "account-access:CreateApplication",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "aws:RequestTag/Environment": "Production"
    },
    "ForAllValues:StringEquals": {
      "aws:TagKeys": [
        "Environment"
      ]
    }
  }
}
```

Facultatif : pour gérer et enregistrer les administrateurs délégués pour le gestionnaire d'accès au compte, ajoutez :

```
{
  "Sid": "ManageDelegatedAdministrator",
  "Effect": "Allow",
  "Action": [
    "organizations:DeregisterDelegatedAdministrator",
    "organizations:RegisterDelegatedAdministrator"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "organizations:ServicePrincipal": "account-access.amazonaws.com"
    }
  }
}
```

Pour attribuer des rôles IAM aux utilisateurs et groupes de votre personnel à partir du compte de gestion de l' AWS organisation sans administrer les paramètres du gestionnaire d'accès au compte, vous devez disposer des autorisations suivantes :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AAMManageEntitlementsOnly",
      "Effect": "Allow",
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteEntitlement",
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements"
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
      "Action": "account-access:ListApplications",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": "sso:ListInstances",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityStoreActions",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
        "sso-directory:DescribeGroup",
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
        "sso-directory:ListGroupsForUser",
        "sso-directory:ListMembersInGroup",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
      ],
```

```
      "Resource": "*"
    },
    {
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

Administration du gestionnaire d'accès au compte dans un compte d'administration délégué

Pour administrer les paramètres du gestionnaire d'accès aux comptes à partir d'un compte d'administration délégué, vous devez disposer des autorisations suivantes :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AAMDelegatedAdminManagement",
      "Effect": "Allow",
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteEntitlement",
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements",
        "account-access:ListTagsForResource",
        "account-access:TagResource",
        "account-access:UntagResource"
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
```

```
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
      "Action": "account-access:ListApplications",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": [
        "sso:DescribeApplication",
        "sso:DescribeInstance",
        "sso:DescribeRegion",
        "sso:ListInstances"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityStoreActions",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
        "sso-directory:DescribeGroup",
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
        "sso-directory:ListGroupsForUser",
        "sso-directory:ListMembersInGroup",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
```

```
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

Pour attribuer des rôles IAM à des utilisateurs et à des groupes de personnel à partir d'un compte d'administration délégué sans administrer les paramètres du gestionnaire d'accès au compte, vous devez disposer des autorisations suivantes :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AAMManageEntitlementsOnly",
      "Effect": "Allow",
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteEntitlement",
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements"
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
      "Action": "account-access:ListApplications",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": "sso:ListInstances",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityStoreActions",
```

```
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
        "sso-directory:DescribeGroup",
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
        "sso-directory:ListGroupsForUser",
        "sso-directory:ListMembersInGroup",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

Protection des données dans le gestionnaire d'accès au compte

Cette section complète les informations fournies dans la section Protection des données Gestion des identités et des accès AWS avec des informations supplémentaires spécifiques au gestionnaire d'accès au compte.

Utilisation des données des utilisateurs et des groupes d'IAM Identity Center

Le gestionnaire d'accès aux comptes utilise les données des utilisateurs et des groupes provenant d'IAM Identity Center pour gérer les attributions de comptes. Cependant, le gestionnaire d'accès au compte stocke et enregistre uniquement les identifiants AWS générés (ID utilisateur et ID de groupe). Il ne stocke ni n'enregistre les noms d'utilisateur, les adresses e-mail ou autres attributs personnels.

Enregistrement et surveillance dans le gestionnaire d'accès aux comptes

Le gestionnaire d'accès aux comptes utilise son propre espace de noms d'API (`account-access`) et enregistre les appels d'API. CloudTrail

Pour plus d'informations à ce sujet CloudTrail, consultez le Guide de AWS CloudTrail l'utilisateur.

Quotas et limites dans le gestionnaire d'accès aux comptes

Les tableaux suivants décrivent les quotas au sein du gestionnaire d'accès au compte. Les demandes d'augmentation de quota doivent provenir d'un compte de gestion ou d'administrateur délégué. Pour augmenter un quota, voir Demande d'augmentation de quota.

Quota d'assignation de groupe

Ressource	Quota par défaut	Peut être augmenté
Nombre maximum de groupes affectés à un rôle IAM	20	Oui

Limites d'accélérateur

Ressource	Quota par défaut	Peut être augmenté
API d'accès au compte	L'API d'accès au compte a une limite d'accélération collective de 20 transactions par seconde (TPS). Pour les API de lecture, vous pouvez	Oui

Ressource	Quota par défaut	Peut être augmenté
	ouvrir un dossier d'assistance pour demander une augmentation de la limite. Les opérations d'écriture de l'API ont une limite de 15 appels asynchrones en attente. Cette limite ne peut pas être augmentée.	

Exemples de code pour l'utilisation d'IAM AWS Kits SDK

Les exemples de code suivants montrent comment utiliser IAM avec un kit de développement AWS logiciel (SDK).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit de développement logiciel (SDK).

Exemples de code

- Exemples de code pour l'utilisation d'IAM AWS Kits SDK
  - Exemples de base pour l'utilisation d'IAM AWS Kits SDK
    - Bonjour IAM
    - Apprenez les bases de l'IAM avec un AWS Kit SDK
    - Actions destinées à l'utilisation d'IAM AWS Kits SDK
      - Utilisation de AddClientIdToOpenIdConnectProvider avec une CLI
      - Utilisation de AddRoleToInstanceProfile avec une CLI
      - Utilisation de AddUserToGroup avec une CLI
      - Utilisation de AttachGroupPolicy avec une CLI
      - À utiliser AttachRolePolicyavec un AWS SDK ou CLI
      - À utiliser AttachUserPolicyavec un AWS SDK ou CLI
      - Utilisation de ChangePassword avec une CLI
      - À utiliser CreateAccessKeyavec un AWS SDK ou CLI
      - À utiliser CreateAccountAliasavec un AWS SDK ou CLI
      - À utiliser CreateGroupavec un AWS SDK ou CLI
      - À utiliser CreateInstanceProfileavec un AWS SDK ou CLI
      - Utilisation de CreateLoginProfile avec une CLI
      - Utilisation de CreateOpenIdConnectProvider avec une CLI
      - À utiliser CreatePolicyavec un AWS SDK ou CLI
      - À utiliser CreatePolicyVersionavec un AWS SDK ou CLI
      - À utiliser CreateRoleavec un AWS SDK ou CLI

- Utilisez CreateSAMLProvider avec un AWS SDK ou CLI
- À utiliser CreateServiceLinkedRoleavec un AWS SDK ou CLI
- À utiliser CreateUseravec un AWS SDK ou CLI
- Utilisation de CreateVirtualMfaDevice avec une CLI
- Utilisation de DeactivateMfaDevice avec une CLI
- À utiliser DeleteAccessKeyavec un AWS SDK ou CLI
- À utiliser DeleteAccountAliasavec un AWS SDK ou CLI
- Utilisation de DeleteAccountPasswordPolicy avec une CLI
- À utiliser DeleteGroupavec un AWS SDK ou CLI
- Utilisation de DeleteGroupPolicy avec une CLI
- À utiliser DeleteInstanceProfileavec un AWS SDK ou CLI
- Utilisation de DeleteLoginProfile avec une CLI
- Utilisation de DeleteOpenIdConnectProvider avec une CLI
- À utiliser DeletePolicyavec un AWS SDK ou CLI
- À utiliser DeletePolicyVersionavec un AWS SDK ou CLI
- À utiliser DeleteRoleavec un AWS SDK ou CLI
- Utilisation de DeleteRolePermissionsBoundary avec une CLI
- À utiliser DeleteRolePolicyavec un AWS SDK ou CLI
- Utilisez DeleteSAMLProvider avec un AWS SDK ou CLI
- À utiliser DeleteServerCertificateavec un AWS SDK ou CLI
- À utiliser DeleteServiceLinkedRoleavec un AWS SDK ou CLI
- Utilisation de DeleteSigningCertificate avec une CLI
- À utiliser DeleteUseravec un AWS SDK ou CLI
- Utilisation de DeleteUserPermissionsBoundary avec une CLI
- À utiliser DeleteUserPolicyavec un AWS SDK ou CLI
- Utilisation de DeleteVirtualMfaDevice avec une CLI
- Utilisation de DetachGroupPolicy avec une CLI
- À utiliser DetachRolePolicyavec un AWS SDK ou CLI
- À utiliser DetachUserPolicyavec un AWS SDK ou CLI
- Utilisation de EnableMfaDevice avec une CLI

- À utiliser GenerateCredentialReportavec un AWS SDK ou CLI
- Utilisation de GenerateServiceLastAccessedDetails avec une CLI
- À utiliser GetAccessKeyLastUsedavec un AWS SDK ou CLI
- À utiliser GetAccountAuthorizationDetailsavec un AWS SDK ou CLI
- À utiliser GetAccountPasswordPolicyavec un AWS SDK ou CLI
- À utiliser GetAccountSummaryavec un AWS SDK ou CLI
- Utilisation de GetContextKeysForCustomPolicy avec une CLI
- Utilisation de GetContextKeysForPrincipalPolicy avec une CLI
- À utiliser GetCredentialReportavec un AWS SDK ou CLI
- Utilisation de GetGroup avec une CLI
- Utilisation de GetGroupPolicy avec une CLI
- Utilisation de GetInstanceProfile avec une CLI
- Utilisation de GetLoginProfile avec une CLI
- Utilisation de GetOpenIdConnectProvider avec une CLI
- À utiliser GetPolicyavec un AWS SDK ou CLI
- À utiliser GetPolicyVersionavec un AWS SDK ou CLI
- À utiliser GetRoleavec un AWS SDK ou CLI
- Utilisation de GetRolePolicy avec une CLI
- Utilisation de GetSamlProvider avec une CLI
- À utiliser GetServerCertificateavec un AWS SDK ou CLI
- Utilisation de GetServiceLastAccessedDetails avec une CLI
- Utilisation de GetServiceLastAccessedDetailsWithEntities avec une CLI
- À utiliser GetServiceLinkedRoleDeletionStatusavec un AWS SDK ou CLI
- À utiliser GetUseravec un AWS SDK ou CLI
- Utilisation de GetUserPolicy avec une CLI
- À utiliser ListAccessKeysavec un AWS SDK ou CLI
- À utiliser ListAccountAliasesavec un AWS SDK ou CLI
- Utilisation de ListAttachedGroupPolicies avec une CLI
- À utiliser ListAttachedRolePoliciesavec un AWS SDK ou CLI
- Utilisation de ListAttachedUserPolicies avec une CLI

- Utilisation de ListEntitiesForPolicy avec une CLI
- Utilisation de ListGroupPolicies avec une CLI
- À utiliser ListGroupsavec un AWS SDK ou CLI
- Utilisation de ListGroupsForUser avec une CLI
- Utilisation de ListInstanceProfiles avec une CLI
- Utilisation de ListInstanceProfilesForRole avec une CLI
- Utilisation de ListMfaDevices avec une CLI
- Utilisation de ListOpenIdConnectProviders avec une CLI
- À utiliser ListPoliciesavec un AWS SDK ou CLI
- À utiliser ListPolicyVersionsavec un AWS SDK ou CLI
- À utiliser ListRolePoliciesavec un AWS SDK ou CLI
- Utilisation de ListRoleTags avec une CLI
- À utiliser ListRolesavec un AWS SDK ou CLI
- Utilisez ListSamlProviders avec un AWS SDK ou CLI
- À utiliser ListServerCertificatesavec un AWS SDK ou CLI
- Utilisation de ListSigningCertificates avec une CLI
- À utiliser ListUserPoliciesavec un AWS SDK ou CLI
- Utilisation de ListUserTags avec une CLI
- À utiliser ListUsersavec un AWS SDK ou CLI
- Utilisation de ListVirtualMfaDevices avec une CLI
- Utilisation de PutGroupPolicy avec une CLI
- Utilisation de PutRolePermissionsBoundary avec une CLI
- À utiliser PutRolePolicyavec un AWS SDK ou CLI
- Utilisation de PutUserPermissionsBoundary avec une CLI
- À utiliser PutUserPolicyavec un AWS SDK ou CLI
- Utilisation de RemoveClientIdFromOpenIdConnectProvider avec une CLI
- Utilisation de RemoveRoleFromInstanceProfile avec une CLI
- Utilisation de RemoveUserFromGroup avec une CLI
- Utilisation de ResyncMfaDevice avec une CLI
- À utiliser SetDefaultPolicyVersionavec un AWS SDK ou CLI

- Utilisation de TagRole avec une CLI
- Utilisation de TagUser avec une CLI
- Utilisation de UntagRole avec une CLI
- Utilisation de UntagUser avec une CLI
- À utiliser UpdateAccessKeyavec un AWS SDK ou CLI
- Utilisation de UpdateAccountPasswordPolicy avec une CLI
- Utilisation de UpdateAssumeRolePolicy avec une CLI
- Utilisation de UpdateGroup avec une CLI
- Utilisation de UpdateLoginProfile avec une CLI
- Utilisation de UpdateOpenIdConnectProviderThumbprint avec une CLI
- Utilisation de UpdateRole avec une CLI
- Utilisation de UpdateRoleDescription avec une CLI
- Utilisation de UpdateSamlProvider avec une CLI
- À utiliser UpdateServerCertificateavec un AWS SDK ou CLI
- Utilisation de UpdateSigningCertificate avec une CLI
- À utiliser UpdateUseravec un AWS SDK ou CLI
- À utiliser UploadServerCertificateavec un AWS SDK ou CLI
- Utilisation de UploadSigningCertificate avec une CLI

- Scénarios d'utilisation d'IAM AWS Kits SDK
  - Créez et gérez un service résilient à l'aide d'un AWS Kit SDK
  - Configuration de la connectivité des services de conteneurs
  - Création d'une tâche de conteneur pour le type de lancement sans serveur
  - Création d'une API REST avec intégration de proxy de fonctions
  - Créez des utilisateurs IAM en lecture seule et en lecture-écriture à l'aide d'un AWS Kit SDK
  - Création d'un service de conteneur pour les instances de machines virtuelles
  - Création d'un espace de travail de surveillance géré
  - Création d'un tableau de bord de surveillance avec le nom de la fonction comme variable
  - Création de votre première fonction sans serveur
  - Commencez avec les entrepôts de données sans serveur à l'aide de la CLI
  - Commencer à gérer la configuration

- Débuter avec la protection des appareils liés à l'Internet des objets
- Commencer à utiliser les magasins de fonctionnalités d'apprentissage automatique
- Commencer à utiliser les clusters Kubernetes gérés
- Débuter avec le streaming géré
- Commencer à utiliser des clusters d'entrepôts de données provisionnés
- Commencer à orchestrer les flux de travail
- Gérez les clés d'accès IAM à l'aide d'un AWS Kit SDK
- Gérez les politiques IAM à l'aide d'un AWS Kit SDK
- Gérez les rôles IAM à l'aide d'un AWS Kit SDK
- Gérez votre compte IAM à l'aide d'un AWS Kit SDK
- Transférer les secrets codés en dur vers un stockage secret sécurisé
- Permet le Optimiseur de calcul AWS Fonction d'automatisation pour appliquer les actions recommandées
- Politique pour activer l’automatisation dans toute votre organisation
- Politique pour activer l’automatisation pour votre compte
- Politique pour accorder un accès complet à l’automatisation de l’optimiseur de calcul au compte de gestion d’une organisation
- Politique visant à accorder un accès complet à Compute Optimizer Automation en mode autonome AWS comptes
- Politique pour accorder un accès en lecture seule à l’automatisation de l’optimiseur de calcul au compte de gestion d’une organisation
- Politique visant à accorder un accès en lecture seule à Compute Optimizer Automation en mode autonome AWS comptes
- Politique pour accorder les autorisations de rôle lié à un service pour l’automatisation de l’optimisation du calcul
- Restaurer une version d'une politique IAM à l'aide d'un AWS Kit SDK
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts
- Configurer le contrôle Attribute-Based d'accès pour DynamoDB à l'aide de AWS Command Line Interface v2
- Configuration de la gestion des systèmes

- Utilisation de variables de propriété dans les tableaux de bord de surveillance pour surveiller plusieurs fonctions sans serveur
- Travaillez avec DynamoDB Streams et utilisez Time-to-Live AWS Command Line Interface v2
- Utilisez l'API IAM Policy Builder à l'aide d'un AWS Kit SDK

- Exemples de code pour AWS STS utilisant AWS Kits SDK
  - Exemples de base pour AWS STS utilisant AWS Kits SDK
    - Actions pour AWS STS utilisant AWS Kits SDK
      - À utiliser AssumeRoleavec un AWS SDK ou CLI
      - Utilisation de AssumeRoleWithWebIdentity avec une CLI
      - Utilisation de DecodeAuthorizationMessage avec une CLI
      - Utilisation de GetFederationToken avec une CLI
      - À utiliser GetSessionTokenavec un AWS SDK ou CLI
    - Scénarios pour AWS STS utilisant AWS Kits SDK
      - Assumez un rôle IAM qui nécessite un jeton MFA avec AWS STS à l'aide d'un AWS Kit SDK
      - Configuration de la connectivité des services de conteneurs
      - Construisez une URL avec AWS STS pour les utilisateurs fédérés utilisant un AWS Kit SDK
      - Création d'une tâche de conteneur pour le type de lancement sans serveur
      - Création d'une API REST avec intégration de proxy de fonctions
      - Création d'un service de conteneur pour les instances de machines virtuelles
      - Création d'un espace de travail de surveillance géré
      - Création d'un tableau de bord de surveillance avec le nom de la fonction comme variable
      - Obtenez un jeton de session qui nécessite un jeton MFA avec AWS STS à l'aide d'un AWS Kit SDK
      - Commencer à gérer la configuration
      - Commencer à utiliser les registres de conteneurs
      - Débuter avec la messagerie de l'Internet des objets
      - Commencer à utiliser les magasins de fonctionnalités d'apprentissage automatique
      - Commencer à utiliser les clusters Kubernetes gérés
      - Débuter avec le streaming géré
      - Commencer à utiliser les notifications push

- Commencer à utiliser les moteurs de recherche et d'analyse
- Commencer à utiliser les pare-feux pour applications Web
- Transférer les secrets codés en dur vers un stockage secret sécurisé
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts
- Configuration de la gestion des systèmes

Exemples de code pour l'utilisation d'IAM AWS Kits SDK

Les exemples de code suivants montrent comment utiliser IAM avec un kit de développement AWS logiciel (SDK).

Les principes de base sont des exemples de code qui vous montrent comment effectuer les opérations essentielles au sein d'un service.

Les actions sont des extraits de code de programmes plus larges et doivent être exécutées dans leur contexte. Alors que les actions vous indiquent comment appeler des fonctions de service individuelles, vous pouvez les voir en contexte dans leurs scénarios associés.

Les scénarios sont des exemples de code qui vous montrent comment accomplir des tâches spécifiques en appelant plusieurs fonctions au sein d'un même service ou combinés à d'autres Services AWS.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

IAM

- Exemples de base pour l'utilisation d'IAM AWS Kits SDK
  - Bonjour IAM
  - Apprenez les bases de l'IAM avec un AWS Kit SDK
  - Actions destinées à l'utilisation d'IAM AWS Kits SDK
    - Utilisation de AddClientIdToOpenIdConnectProvider avec une CLI
    - Utilisation de AddRoleToInstanceProfile avec une CLI
    - Utilisation de AddUserToGroup avec une CLI
    - Utilisation de AttachGroupPolicy avec une CLI

- À utiliser AttachRolePolicyavec un AWS SDK ou CLI
- À utiliser AttachUserPolicyavec un AWS SDK ou CLI
- Utilisation de ChangePassword avec une CLI
- À utiliser CreateAccessKeyavec un AWS SDK ou CLI
- À utiliser CreateAccountAliasavec un AWS SDK ou CLI
- À utiliser CreateGroupavec un AWS SDK ou CLI
- À utiliser CreateInstanceProfileavec un AWS SDK ou CLI
- Utilisation de CreateLoginProfile avec une CLI
- Utilisation de CreateOpenIdConnectProvider avec une CLI
- À utiliser CreatePolicyavec un AWS SDK ou CLI
- À utiliser CreatePolicyVersionavec un AWS SDK ou CLI
- À utiliser CreateRoleavec un AWS SDK ou CLI
- Utilisez CreateSAMLProvider avec un AWS SDK ou CLI
- À utiliser CreateServiceLinkedRoleavec un AWS SDK ou CLI
- À utiliser CreateUseravec un AWS SDK ou CLI
- Utilisation de CreateVirtualMfaDevice avec une CLI
- Utilisation de DeactivateMfaDevice avec une CLI
- À utiliser DeleteAccessKeyavec un AWS SDK ou CLI
- À utiliser DeleteAccountAliasavec un AWS SDK ou CLI
- Utilisation de DeleteAccountPasswordPolicy avec une CLI
- À utiliser DeleteGroupavec un AWS SDK ou CLI
- Utilisation de DeleteGroupPolicy avec une CLI
- À utiliser DeleteInstanceProfileavec un AWS SDK ou CLI
- Utilisation de DeleteLoginProfile avec une CLI
- Utilisation de DeleteOpenIdConnectProvider avec une CLI
- À utiliser DeletePolicyavec un AWS SDK ou CLI
- À utiliser DeletePolicyVersionavec un AWS SDK ou CLI
- À utiliser DeleteRoleavec un AWS SDK ou CLI
- Utilisation de DeleteRolePermissionsBoundary avec une CLI
- À utiliser DeleteRolePolicyavec un AWS SDK ou CLI

- Utilisez DeleteSAMLProvider avec un AWS SDK ou CLI
- À utiliser DeleteServerCertificateavec un AWS SDK ou CLI
- À utiliser DeleteServiceLinkedRoleavec un AWS SDK ou CLI
- Utilisation de DeleteSigningCertificate avec une CLI
- À utiliser DeleteUseravec un AWS SDK ou CLI
- Utilisation de DeleteUserPermissionsBoundary avec une CLI
- À utiliser DeleteUserPolicyavec un AWS SDK ou CLI
- Utilisation de DeleteVirtualMfaDevice avec une CLI
- Utilisation de DetachGroupPolicy avec une CLI
- À utiliser DetachRolePolicyavec un AWS SDK ou CLI
- À utiliser DetachUserPolicyavec un AWS SDK ou CLI
- Utilisation de EnableMfaDevice avec une CLI
- À utiliser GenerateCredentialReportavec un AWS SDK ou CLI
- Utilisation de GenerateServiceLastAccessedDetails avec une CLI
- À utiliser GetAccessKeyLastUsedavec un AWS SDK ou CLI
- À utiliser GetAccountAuthorizationDetailsavec un AWS SDK ou CLI
- À utiliser GetAccountPasswordPolicyavec un AWS SDK ou CLI
- À utiliser GetAccountSummaryavec un AWS SDK ou CLI
- Utilisation de GetContextKeysForCustomPolicy avec une CLI
- Utilisation de GetContextKeysForPrincipalPolicy avec une CLI
- À utiliser GetCredentialReportavec un AWS SDK ou CLI
- Utilisation de GetGroup avec une CLI
- Utilisation de GetGroupPolicy avec une CLI
- Utilisation de GetInstanceProfile avec une CLI
- Utilisation de GetLoginProfile avec une CLI
- Utilisation de GetOpenIdConnectProvider avec une CLI
- À utiliser GetPolicyavec un AWS SDK ou CLI
- À utiliser GetPolicyVersionavec un AWS SDK ou CLI
- À utiliser GetRoleavec un AWS SDK ou CLI
- Utilisation de GetRolePolicy avec une CLI

- Utilisation de GetSamlProvider avec une CLI
- À utiliser GetServerCertificateavec un AWS SDK ou CLI
- Utilisation de GetServiceLastAccessedDetails avec une CLI
- Utilisation de GetServiceLastAccessedDetailsWithEntities avec une CLI
- À utiliser GetServiceLinkedRoleDeletionStatusavec un AWS SDK ou CLI
- À utiliser GetUseravec un AWS SDK ou CLI
- Utilisation de GetUserPolicy avec une CLI
- À utiliser ListAccessKeysavec un AWS SDK ou CLI
- À utiliser ListAccountAliasesavec un AWS SDK ou CLI
- Utilisation de ListAttachedGroupPolicies avec une CLI
- À utiliser ListAttachedRolePoliciesavec un AWS SDK ou CLI
- Utilisation de ListAttachedUserPolicies avec une CLI
- Utilisation de ListEntitiesForPolicy avec une CLI
- Utilisation de ListGroupPolicies avec une CLI
- À utiliser ListGroupsavec un AWS SDK ou CLI
- Utilisation de ListGroupsForUser avec une CLI
- Utilisation de ListInstanceProfiles avec une CLI
- Utilisation de ListInstanceProfilesForRole avec une CLI
- Utilisation de ListMfaDevices avec une CLI
- Utilisation de ListOpenIdConnectProviders avec une CLI
- À utiliser ListPoliciesavec un AWS SDK ou CLI
- À utiliser ListPolicyVersionsavec un AWS SDK ou CLI
- À utiliser ListRolePoliciesavec un AWS SDK ou CLI
- Utilisation de ListRoleTags avec une CLI
- À utiliser ListRolesavec un AWS SDK ou CLI
- Utilisez ListSamlProviders avec un AWS SDK ou CLI
- À utiliser ListServerCertificatesavec un AWS SDK ou CLI
- Utilisation de ListSigningCertificates avec une CLI
- À utiliser ListUserPoliciesavec un AWS SDK ou CLI
- Utilisation de ListUserTags avec une CLI

- À utiliser ListUsersavec un AWS SDK ou CLI
- Utilisation de ListVirtualMfaDevices avec une CLI
- Utilisation de PutGroupPolicy avec une CLI
- Utilisation de PutRolePermissionsBoundary avec une CLI
- À utiliser PutRolePolicyavec un AWS SDK ou CLI
- Utilisation de PutUserPermissionsBoundary avec une CLI
- À utiliser PutUserPolicyavec un AWS SDK ou CLI
- Utilisation de RemoveClientIdFromOpenIdConnectProvider avec une CLI
- Utilisation de RemoveRoleFromInstanceProfile avec une CLI
- Utilisation de RemoveUserFromGroup avec une CLI
- Utilisation de ResyncMfaDevice avec une CLI
- À utiliser SetDefaultPolicyVersionavec un AWS SDK ou CLI
- Utilisation de TagRole avec une CLI
- Utilisation de TagUser avec une CLI
- Utilisation de UntagRole avec une CLI
- Utilisation de UntagUser avec une CLI
- À utiliser UpdateAccessKeyavec un AWS SDK ou CLI
- Utilisation de UpdateAccountPasswordPolicy avec une CLI
- Utilisation de UpdateAssumeRolePolicy avec une CLI
- Utilisation de UpdateGroup avec une CLI
- Utilisation de UpdateLoginProfile avec une CLI
- Utilisation de UpdateOpenIdConnectProviderThumbprint avec une CLI
- Utilisation de UpdateRole avec une CLI
- Utilisation de UpdateRoleDescription avec une CLI
- Utilisation de UpdateSamlProvider avec une CLI
- À utiliser UpdateServerCertificateavec un AWS SDK ou CLI
- Utilisation de UpdateSigningCertificate avec une CLI
- À utiliser UpdateUseravec un AWS SDK ou CLI
- À utiliser UploadServerCertificateavec un AWS SDK ou CLI
- Utilisation de UploadSigningCertificate avec une CLI

- Scénarios d'utilisation d'IAM AWS Kits SDK
  - Créez et gérez un service résilient à l'aide d'un AWS Kit SDK
  - Configuration de la connectivité des services de conteneurs
  - Création d'une tâche de conteneur pour le type de lancement sans serveur
  - Création d'une API REST avec intégration de proxy de fonctions
  - Créez des utilisateurs IAM en lecture seule et en lecture-écriture à l'aide d'un AWS Kit SDK
  - Création d'un service de conteneur pour les instances de machines virtuelles
  - Création d'un espace de travail de surveillance géré
  - Création d'un tableau de bord de surveillance avec le nom de la fonction comme variable
  - Création de votre première fonction sans serveur
  - Commencez avec les entrepôts de données sans serveur à l'aide de la CLI
  - Commencer à gérer la configuration
  - Débuter avec la protection des appareils liés à l'Internet des objets
  - Commencer à utiliser les magasins de fonctionnalités d'apprentissage automatique
  - Commencer à utiliser les clusters Kubernetes gérés
  - Débuter avec le streaming géré
  - Commencer à utiliser des clusters d'entrepôts de données provisionnés
  - Commencer à orchestrer les flux de travail
  - Gérez les clés d'accès IAM à l'aide d'un AWS Kit SDK
  - Gérez les politiques IAM à l'aide d'un AWS Kit SDK
  - Gérez les rôles IAM à l'aide d'un AWS Kit SDK
  - Gérez votre compte IAM à l'aide d'un AWS Kit SDK
  - Transférer les secrets codés en dur vers un stockage secret sécurisé
  - Permet le Optimiseur de calcul AWS Fonction d'automatisation pour appliquer les actions recommandées
  - Politique pour activer l'automatisation dans toute votre organisation
  - Politique pour activer l'automatisation pour votre compte
  - Politique pour accorder un accès complet à l'automatisation de l'optimiseur de calcul au compte de gestion d'une organisation
  - Politique visant à accorder un accès complet à Compute Optimizer Automation en mode autonome AWS comptes

- Politique pour accorder un accès en lecture seule à l'automatisation de l'optimiseur de calcul au compte de gestion d'une organisation
- Politique visant à accorder un accès en lecture seule à Compute Optimizer Automation en mode autonome AWS comptes
- Politique pour accorder les autorisations de rôle lié à un service pour l'automatisation de l'optimisation du calcul
- Restaurer une version d'une politique IAM à l'aide d'un AWS Kit SDK
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts
- Configurer le contrôle Attribute-Based d'accès pour DynamoDB à l'aide de AWS Command Line Interface v2
- Configuration de la gestion des systèmes
- Utilisation de variables de propriété dans les tableaux de bord de surveillance pour surveiller plusieurs fonctions sans serveur
- Travaillez avec DynamoDB Streams et utilisez Time-to-Live AWS Command Line Interface v2
- Utilisez l'API IAM Policy Builder à l'aide d'un AWS Kit SDK

Exemples de base pour l'utilisation d'IAM AWS Kits SDK

Les exemples de code suivants montrent comment utiliser les principes de base Gestion des identités et des accès AWS des AWS SDK.

Exemples

- Bonjour IAM
- Apprenez les bases de l'IAM avec un AWS Kit SDK
- Actions destinées à l'utilisation d'IAM AWS Kits SDK
  - Utilisation de AddClientIdToOpenIdConnectProvider avec une CLI
  - Utilisation de AddRoleToInstanceProfile avec une CLI
  - Utilisation de AddUserToGroup avec une CLI
  - Utilisation de AttachGroupPolicy avec une CLI
  - À utiliser AttachRolePolicyavec un AWS SDK ou CLI
  - À utiliser AttachUserPolicyavec un AWS SDK ou CLI
  - Utilisation de ChangePassword avec une CLI

- À utiliser CreateAccessKeyavec un AWS SDK ou CLI
- À utiliser CreateAccountAliasavec un AWS SDK ou CLI
- À utiliser CreateGroupavec un AWS SDK ou CLI
- À utiliser CreateInstanceProfileavec un AWS SDK ou CLI
- Utilisation de CreateLoginProfile avec une CLI
- Utilisation de CreateOpenIdConnectProvider avec une CLI
- À utiliser CreatePolicyavec un AWS SDK ou CLI
- À utiliser CreatePolicyVersionavec un AWS SDK ou CLI
- À utiliser CreateRoleavec un AWS SDK ou CLI
- Utilisez CreateSAMLProvider avec un AWS SDK ou CLI
- À utiliser CreateServiceLinkedRoleavec un AWS SDK ou CLI
- À utiliser CreateUseravec un AWS SDK ou CLI
- Utilisation de CreateVirtualMfaDevice avec une CLI
- Utilisation de DeactivateMfaDevice avec une CLI
- À utiliser DeleteAccessKeyavec un AWS SDK ou CLI
- À utiliser DeleteAccountAliasavec un AWS SDK ou CLI
- Utilisation de DeleteAccountPasswordPolicy avec une CLI
- À utiliser DeleteGroupavec un AWS SDK ou CLI
- Utilisation de DeleteGroupPolicy avec une CLI
- À utiliser DeleteInstanceProfileavec un AWS SDK ou CLI
- Utilisation de DeleteLoginProfile avec une CLI
- Utilisation de DeleteOpenIdConnectProvider avec une CLI
- À utiliser DeletePolicyavec un AWS SDK ou CLI
- À utiliser DeletePolicyVersionavec un AWS SDK ou CLI
- À utiliser DeleteRoleavec un AWS SDK ou CLI
- Utilisation de DeleteRolePermissionsBoundary avec une CLI
- À utiliser DeleteRolePolicyavec un AWS SDK ou CLI
- Utilisez DeleteSAMLProvider avec un AWS SDK ou CLI
- À utiliser DeleteServerCertificateavec un AWS SDK ou CLI

- À utiliser DeleteServiceLinkedRoleavec un AWS SDK ou CLI

- Utilisation de DeleteSigningCertificate avec une CLI
- À utiliser DeleteUseravec un AWS SDK ou CLI
- Utilisation de DeleteUserPermissionsBoundary avec une CLI
- À utiliser DeleteUserPolicyavec un AWS SDK ou CLI
- Utilisation de DeleteVirtualMfaDevice avec une CLI
- Utilisation de DetachGroupPolicy avec une CLI
- À utiliser DetachRolePolicyavec un AWS SDK ou CLI
- À utiliser DetachUserPolicyavec un AWS SDK ou CLI
- Utilisation de EnableMfaDevice avec une CLI
- À utiliser GenerateCredentialReportavec un AWS SDK ou CLI
- Utilisation de GenerateServiceLastAccessedDetails avec une CLI
- À utiliser GetAccessKeyLastUsedavec un AWS SDK ou CLI
- À utiliser GetAccountAuthorizationDetailsavec un AWS SDK ou CLI
- À utiliser GetAccountPasswordPolicyavec un AWS SDK ou CLI
- À utiliser GetAccountSummaryavec un AWS SDK ou CLI
- Utilisation de GetContextKeysForCustomPolicy avec une CLI
- Utilisation de GetContextKeysForPrincipalPolicy avec une CLI
- À utiliser GetCredentialReportavec un AWS SDK ou CLI
- Utilisation de GetGroup avec une CLI
- Utilisation de GetGroupPolicy avec une CLI
- Utilisation de GetInstanceProfile avec une CLI
- Utilisation de GetLoginProfile avec une CLI
- Utilisation de GetOpenIdConnectProvider avec une CLI
- À utiliser GetPolicyavec un AWS SDK ou CLI
- À utiliser GetPolicyVersionavec un AWS SDK ou CLI
- À utiliser GetRoleavec un AWS SDK ou CLI
- Utilisation de GetRolePolicy avec une CLI
- Utilisation de GetSamlProvider avec une CLI
- À utiliser GetServerCertificateavec un AWS SDK ou CLI
- Utilisation de GetServiceLastAccessedDetails avec une CLI

- Utilisation de GetServiceLastAccessedDetailsWithEntities avec une CLI
- À utiliser GetServiceLinkedRoleDeletionStatusavec un AWS SDK ou CLI
- À utiliser GetUseravec un AWS SDK ou CLI
- Utilisation de GetUserPolicy avec une CLI
- À utiliser ListAccessKeysavec un AWS SDK ou CLI
- À utiliser ListAccountAliasesavec un AWS SDK ou CLI
- Utilisation de ListAttachedGroupPolicies avec une CLI
- À utiliser ListAttachedRolePoliciesavec un AWS SDK ou CLI
- Utilisation de ListAttachedUserPolicies avec une CLI
- Utilisation de ListEntitiesForPolicy avec une CLI
- Utilisation de ListGroupPolicies avec une CLI
- À utiliser ListGroupsavec un AWS SDK ou CLI
- Utilisation de ListGroupsForUser avec une CLI
- Utilisation de ListInstanceProfiles avec une CLI
- Utilisation de ListInstanceProfilesForRole avec une CLI
- Utilisation de ListMfaDevices avec une CLI
- Utilisation de ListOpenIdConnectProviders avec une CLI
- À utiliser ListPoliciesavec un AWS SDK ou CLI
- À utiliser ListPolicyVersionsavec un AWS SDK ou CLI
- À utiliser ListRolePoliciesavec un AWS SDK ou CLI
- Utilisation de ListRoleTags avec une CLI
- À utiliser ListRolesavec un AWS SDK ou CLI
- Utilisez ListSamlProviders avec un AWS SDK ou CLI
- À utiliser ListServerCertificatesavec un AWS SDK ou CLI
- Utilisation de ListSigningCertificates avec une CLI
- À utiliser ListUserPoliciesavec un AWS SDK ou CLI
- Utilisation de ListUserTags avec une CLI
- À utiliser ListUsersavec un AWS SDK ou CLI
- Utilisation de ListVirtualMfaDevices avec une CLI

- Utilisation de PutGroupPolicy avec une CLI

- Utilisation de PutRolePermissionsBoundary avec une CLI
- À utiliser PutRolePolicyavec un AWS SDK ou CLI
- Utilisation de PutUserPermissionsBoundary avec une CLI
- À utiliser PutUserPolicyavec un AWS SDK ou CLI
- Utilisation de RemoveClientIdFromOpenIdConnectProvider avec une CLI
- Utilisation de RemoveRoleFromInstanceProfile avec une CLI
- Utilisation de RemoveUserFromGroup avec une CLI
- Utilisation de ResyncMfaDevice avec une CLI
- À utiliser SetDefaultPolicyVersionavec un AWS SDK ou CLI
- Utilisation de TagRole avec une CLI
- Utilisation de TagUser avec une CLI
- Utilisation de UntagRole avec une CLI
- Utilisation de UntagUser avec une CLI
- À utiliser UpdateAccessKeyavec un AWS SDK ou CLI
- Utilisation de UpdateAccountPasswordPolicy avec une CLI
- Utilisation de UpdateAssumeRolePolicy avec une CLI
- Utilisation de UpdateGroup avec une CLI
- Utilisation de UpdateLoginProfile avec une CLI
- Utilisation de UpdateOpenIdConnectProviderThumbprint avec une CLI
- Utilisation de UpdateRole avec une CLI
- Utilisation de UpdateRoleDescription avec une CLI
- Utilisation de UpdateSamlProvider avec une CLI
- À utiliser UpdateServerCertificateavec un AWS SDK ou CLI
- Utilisation de UpdateSigningCertificate avec une CLI
- À utiliser UpdateUseravec un AWS SDK ou CLI
- À utiliser UploadServerCertificateavec un AWS SDK ou CLI
- Utilisation de UploadSigningCertificate avec une CLI

Bonjour IAM

Les exemples de code suivants montrent comment démarrer avec IAM.

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
namespace IAMActions;

public class HelloIAM
{
    static async Task Main(string[] args)
    {
        // Getting started with AWS Identity and Access Management (IAM). List
        // the policies for the account.
        var iamClient = new AmazonIdentityManagementServiceClient();

        var listPoliciesPaginator = iamClient.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        Console.WriteLine("Here are the policies defined for your account:\n");
        policies.ForEach(policy =>
        {
            Console.WriteLine($"Created:
 {policy.CreateDate}\t{policy.PolicyName}\t{policy.Description}");
        });
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Code pour le fichier CMakeLists.txt CMake.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS iam)

# Set this project's name.
project("hello_iam")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
    # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.
```

```
    # set(BIN_SUB_DIR "/Debug") # if you are building from the command line you
 may need to uncomment this
    # and set the proper subdirectory to the executables' location.

    AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_iam.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Code pour le fichier source iam.cpp.

```
#include <aws/core/Aws.h>
#include <aws/iam/IAMClient.h>
#include <aws/iam/model/ListPoliciesRequest.h>
#include <iostream>
#include <iomanip>

/*
 *  A "Hello IAM" starter application which initializes an AWS Identity and
 Access Management (IAM) client
 *  and lists the IAM policies.
 *
 *  main function
 *
 *  Usage: 'hello_iam'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        const Aws::String DATE_FORMAT("%Y-%m-%d");
        Aws::Client::ClientConfiguration clientConfig;
```

```
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::IAM::IAMClient iamClient(clientConfig);
        Aws::IAM::Model::ListPoliciesRequest request;

        bool done = false;
        bool header = false;
        while (!done) {
            auto outcome = iamClient.ListPolicies(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Failed to list iam policies: " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = 1;
                break;
            }

            if (!header) {
                std::cout << std::left << std::setw(55) << "Name" <<
                          std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                          std::setw(64) << "Description" << std::setw(12) <<
                          "CreateDate" << std::endl;
                header = true;
            }

            const auto &policies = outcome.GetResult().GetPolicies();
            for (const auto &policy: policies) {
                std::cout << std::left << std::setw(55) <<
                          policy.GetPolicyName() << std::setw(30) <<
                          policy.GetPolicyId() << std::setw(80) <<
policy.GetArn() <<
                          std::setw(64) << policy.GetDescription() <<
std::setw(12) <<
                          policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
<<
                          std::endl;
            }

            if (outcome.GetResult().GetIsTruncated()) {
                request.SetMarker(outcome.GetResult().GetMarker());
            } else {
                done = true;
            }
        }
```

```
    }


    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK pour C++ API.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
package main

import (
 "context"
 "fmt"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/iam"
)

// main uses the AWS SDK for Go (v2) to create an AWS Identity and Access
 Management (IAM)
// client and list up to 10 policies in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
```

```
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 iamClient := iam.NewFromConfig(sdkConfig)
 const maxPols = 10
 fmt.Printf("Let's list up to %v policies for your account.\n", maxPols)
 result, err := iamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPols),
 })
 if err != nil {
  fmt.Printf("Couldn't list policies for your account. Here's why: %v\n", err)
  return
 }
 if len(result.Policies) == 0 {
  fmt.Println("You don't have any policies!")
 } else {
  for _, policy := range result.Policies {
   fmt.Printf("\t%v\n", *policy.PolicyName)
  }
 }
}
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK pour Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.regions.Region;
```

```
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.ListPoliciesResponse;
import software.amazon.awssdk.services.iam.model.Policy;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class HelloIAM {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listPolicies(iam);
    }

    public static void listPolicies(IamClient iam) {
        ListPoliciesResponse response = iam.listPolicies();
        List<Policy> polList = response.policies();
        polList.forEach(policy -> {
            System.out.println("Policy Name: " + policy.policyName());
        });
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { IAMClient, paginateListPolicies } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listLocalPolicies = async () => {
  /**
   * In v3, the clients expose paginateOperationName APIs that are written using
 async generators so that you can use async iterators in a for await..of loop.
   * https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/index.html#paginators
   */
  const paginator = paginateListPolicies(
    { client, pageSize: 10 },
    // List only customer managed policies.
    { Scope: "Local" },
  );

  console.log("IAM policies defined in your account:");
  let policyCount = 0;
  for await (const page of paginator) {
    if (page.Policies) {
      for (const policy of page.Policies) {
        console.log(`${policy.PolicyName}`);
        policyCount++;
      }
    }
  }
  console.log(`Found ${policyCount} policies.`);
};
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK pour JavaScript API.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import boto3


def main():
    """
    Lists the managed policies in your AWS account using the AWS SDK for Python
 (Boto3).
    """
    iam = boto3.client("iam")

    try:
        # Get a paginator for the list_policies operation
        paginator = iam.get_paginator("list_policies")

        # Iterate through the pages of results
        for page in paginator.paginate(Scope="All", OnlyAttached=False):
            for policy in page["Policies"]:
                print(f"Policy name: {policy['PolicyName']}")
                print(f"  Policy ARN: {policy['Arn']}")
    except boto3.exceptions.BotoCoreError as e:
        print(f"Encountered an error while listing policies: {e}")


if __name__ == "__main__":
    main()
```

- Pour plus de détails sur l'API, consultez ListPolicieslе AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
require 'aws-sdk-iam'
require 'logger'

# IAMManager is a class responsible for managing IAM operations
# such as listing all IAM policies in the current AWS account.
class IAMManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all IAM policies in the current AWS account.
  def list_policies
    @logger.info('Here are the IAM policies in your account:')

    paginator = @client.list_policies
    policies = []

    paginator.each_page do |page|
      policies.concat(page.policies)
    end

    if policies.empty?
      @logger.info("You don't have any IAM policies.")
    else
      policies.each do |policy|
        @logger.info("- #{policy.policy_name}")
      end
    end
  end
end
```

```
if $PROGRAM_NAME == __FILE__
  iam_client = Aws::IAM::Client.new
  manager = IAMManager.new(iam_client)
  manager.list_policies
end
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Depuis src/bin /hello.rs.

```
use aws_sdk_iam::error::SdkError;
use aws_sdk_iam::operation::list_policies::ListPoliciesError;
use clap::Parser;

const PATH_PREFIX_HELP: &str = "The path prefix for filtering the results.";

#[derive(Debug, clap::Parser)]
#[command(about)]
struct HelloScenarioArgs {
    #[arg(long, default_value="/", help=PATH_PREFIX_HELP)]
    pub path_prefix: String,
}

#[tokio::main]
async fn main() -> Result<(), Box<SdkError<ListPoliciesError>>> {
    let sdk_config = aws_config::load_from_env().await;
```

```
    let client = aws_sdk_iam::Client::new(&sdk_config);

    let args = HelloScenarioArgs::parse();

    iam_service::list_policies(client, args.path_prefix).await?;

    Ok(())
}
```

Depuis src/iam -service-lib.rs.

```
pub async fn list_policies(
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
        .try_collect()
        .await?;

    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- Pour plus de détails sur l'API, voir ListPolicieslа section de référence de l'API AWS SDK for Rust.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Apprenez les bases de l'IAM avec un AWS Kit SDK

Les exemples de code suivants montrent comment créer un utilisateur et assumer un rôle.

> **Warning**
>
> Afin d'éviter les risques de sécurité, n'employez pas les utilisateurs IAM pour l'authentification lorsque vous développez des logiciels spécialisés ou lorsque vous travaillez avec des données réelles. Préférez la fédération avec un fournisseur d'identité tel que AWS IAM Identity Center.

- Créer un utilisateur sans autorisation.
- Créer un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3 pour le compte.
- Ajouter une politique pour permettre à l'utilisateur d'assumer le rôle.
- Assumez le rôle et répertorier les compartiments S3 à l'aide d'informations d'identification temporaires, puis nettoyez les ressources.

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
```

```
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
```

```
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }



    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }



    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
```

```
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
```

```
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
```

```
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
 { UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

```
   /// <summary>
   /// Delete an IAM user policy.
   /// </summary>
   /// <param name="policyName">The name of the IAM policy to delete.</param>
   /// <param name="userName">The username of the IAM user.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
   {
       var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }


   /// <summary>
   /// Detach an IAM policy from an IAM role.
   /// </summary>
   /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
   /// <param name="roleName">The name of the IAM role.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
   {
       var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
       {
           PolicyArn = policyArn,
           RoleName = roleName,
       });

       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }


   /// <summary>
   /// Gets the IAM password policy for an AWS account.
   /// </summary>
   /// <returns>The PasswordPolicy for the AWS account.</returns>
   public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
   {
```

```
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
```

```
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }
```

```
        return groups;
    }


    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }
```

```
   /// <summary>
   /// List IAM roles.
   /// </summary>
   /// <returns>A list of IAM roles.</returns>
   public async Task<List<Role>> ListRolesAsync()
   {
       var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
       var roles = new List<Role>();

       await foreach (var response in listRolesPaginator.Responses)
       {
           roles.AddRange(response.Roles);
       }

       return roles;
   }


   /// <summary>
   /// List SAML authentication providers.
   /// </summary>
   /// <returns>A list of SAML providers.</returns>
   public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
   {
       var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
       return response.SAMLProviderList;
   }


   /// <summary>
   /// List IAM users.
   /// </summary>
   /// <returns>A list of IAM users.</returns>
   public async Task<List<User>> ListUsersAsync()
   {
       var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
       var users = new List<User>();

       await foreach (var response in listUsersPaginator.Responses)
       {
```

```
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
 policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
```

```
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
 accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}



using Microsoft.Extensions.Configuration;

namespace IAMBasics;
```

```
public class IAMBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();


        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();
```

```
        // First create a user. By default, the new user has
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;

        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
              "\"Effect\": \"Allow\"," +
              "\"Principal\": {" +
              $" \"AWS\": \"{userArn}\"" +
              "}," +
              "\"Action\": \"sts:AssumeRole\"" +
          "}]" +
        "}";

        // Permissions to list all buckets.
        string policyDocument = "{" +
            "\"Version\": \"2012-10-17\"," +
            " \"Statement\" : [{" +
                " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
                " \"Effect\" : \"Allow\"," +
                " \"Resource\" : \"*\"" +
            "}]" +
        "}";

        // Create an AccessKey for the user.
        uiWrapper.DisplayTitle("Create access key");
        Console.WriteLine("Now let's create an access key for the new user.");
        var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);

        var accessKeyId = accessKey.AccessKeyId;
        var secretAccessKey = accessKey.SecretAccessKey;
```

```
        Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");

        Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
        var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

        // Now try listing the Amazon Simple Storage Service (Amazon S3)
        // buckets. This should fail at this point because the user doesn't
        // have permissions to perform this task.
        uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
        Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
        var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
        var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
        var buckets = await s3Wrapper.ListMyBucketsAsync();

        Console.WriteLine(buckets is null
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");

        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
```

```
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);

        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);

        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        // Wait for the new credentials to become valid.
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");

        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);

        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);

        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName} created:
{bucket.CreationDate}");
            });
        }
```

```
        uiWrapper.PressEnter();

        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
 complete.");
        Console.WriteLine("Please wait while we clean up the resources we
 created.");

        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);

        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);

        uiWrapper.PressEnter();

        Console.WriteLine("All done cleaning up our resources. Thank you for your
 patience.");
    }
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
```

```
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
 { BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }
```

```
    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
{ BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
stsService)
```

```
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
 full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
 creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
```

```
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
```

```
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour .NET .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy

- PutUserPolicy

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
  echo
  echo "This demo will create an IAM user, create an IAM role, and apply the role
 to the user."
  echo_repeat "*" 88
```

```
  echo

  echo -n "Enter a name for a new IAM user: "
  get_input
  user_name=$get_input_result

  local user_arn
  user_arn=$(iam_create_user -u "$user_name")

  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created demo IAM user named $user_name"
  else
    errecho "$user_arn"
    errecho "The user failed to create. This demo will exit."
    return 1
  fi

  local access_key_response
  access_key_response=$(iam_create_user_access_key -u "$user_name")
  # shellcheck disable=SC2181
  if [[ ${?} != 0 ]]; then
    errecho "The access key failed to create. This demo will exit."
    clean_up "$user_name"
    return 1
  fi

  IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
  local key_name=${access_key_values[0]}
  local key_secret=${access_key_values[1]}

  echo "Created access key named $key_name"

  echo "Wait 10 seconds for the user to be ready."
  sleep 10
  echo_repeat "*" 88
  echo

  local iam_role_name
  iam_role_name=$(generate_random_name "test-role")
  echo "Creating a role named $iam_role_name with user $user_name as the
principal."

  local assume_role_policy_document="{
```

```
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
        \"Principal\": {\"AWS\": \"$user_arn\"},
        \"Action\": \"sts:AssumeRole\"
        }]
    }"

  local role_arn
  role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
  else
    errecho "The role failed to create. This demo will exit."
    clean_up "$user_name" "$key_name"
    return 1
  fi

  local policy_name
  policy_name=$(generate_random_name "test-policy")
  local policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]}"

  local policy_arn
  policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created  IAM policy named $policy_name"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name"
    return 1
  fi

  if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
    echo "Attached policy $policy_arn to role $iam_role_name"
  else
```

```
    errecho "The policy failed to attach."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
    return 1
  fi

  local assume_role_policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"$role_arn\"}]}"

  local assume_role_policy_name
  assume_role_policy_name=$(generate_random_name "test-assume-role-")

  # shellcheck disable=SC2181
  local assume_role_policy_arn
  assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
    return 1
  fi

  echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
  sleep 10
  echo_repeat "*" 88
  echo

  echo "Try to list buckets without the new user assuming the role."
  echo_repeat "*" 88
  echo

  # Set the environment variables for the created user.
  # bashsupport disable=BP2001
  export AWS_ACCESS_KEY_ID=$key_name
  # bashsupport disable=BP2001
  export AWS_SECRET_ACCESS_KEY=$key_secret
```

```
 local buckets
 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. This should not have
happened."
 else
   errecho "Because the role with permissions has not been assumed, listing
buckets failed."
 fi

 echo
 echo_repeat "*" 88
 echo "Now assume the role $iam_role_name and list the buckets."
 echo_repeat "*" 88
 echo

 local credentials

 credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Assumed role $iam_role_name"
 else
   errecho "Failed to assume role."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 IFS=$'\t ' read -r -a credentials <<<"$credentials"

 export AWS_ACCESS_KEY_ID=${credentials[0]}
 export AWS_SECRET_ACCESS_KEY=${credentials[1]}
 # bashsupport disable=BP2001
 export AWS_SESSION_TOKEN=${credentials[2]}

 buckets=$(s3_list_buckets)
```

```
  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. Listing buckets
 succeeded because of "
    echo "the assumed role."
  else
    errecho "Failed to list buckets. This should not happen."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    export AWS_SESSION_TOKEN=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
 "$policy_arn" "$assume_role_policy_arn"
    return 1
  fi

  local result=0
  export AWS_ACCESS_KEY_ID=""
  export AWS_SECRET_ACCESS_KEY=""

  echo
  echo_repeat "*" 88
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

Fonctions IAM utilisées dans ce scénario.

```
################################################################################
```

```
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
```

```
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi
```

```
  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
```

```
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
```

```
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
```

```
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-role \
  --role-name "$role_name" \
  --assume-role-policy-document "$policy_document" \
  --output text \
  --query Role.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-role operation failed.\n$response"
```

```
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-policy \
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
```

```
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi
```

```
  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
```

```
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam detach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
```

```
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an AWS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
```

```
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
```

```
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an AWS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")
```

```
  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an AWS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
```

```
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    k) access_key="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

if [[ -z "$access_key" ]]; then
  errecho "ERROR: You must provide an access key with the -k parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Username:   $user_name"
iecho "    Access key:   $access_key"
iecho ""

response=$(aws iam delete-access-key \
  --user-name "$user_name" \
  --access-key-id "$access_key")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
  return 1
fi
```

```
  iecho "delete-access-key response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
```

```
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy

- CreateAccessKey
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
namespace AwsDoc {
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
                                          const Aws::IAM::Model::Policy &policy);
```

```
    }

    static const int LIST_BUCKETS_WAIT_SEC = 20;

    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);

        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
 std::endl;
        }
```

```
        user = outcome.GetResult().GetUser();
    }

    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;

        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);

        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());
```

```
        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
    }

    // 3. Create an IAM policy.
    {
        Aws::IAM::Model::CreatePolicyRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String policyName = "iam-demo-policy-" +
                                 Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetPolicyName(policyName);

        // Build IAM policy document.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");
        jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
        jsonStatement.WithString("Resource", "arn:aws:s3:::*");
```

```
        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                  << std::endl;

        // Set IAM policy document as JSON string.
        request.SetPolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a policy with name, " <<
policyName <<
                      "." << std::endl;
        }

        policy = outcome.GetResult().GetPolicy();
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleSessionName(roleSessionName);
```

```
        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;

                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
        }

        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }


    // 5. List objects in the bucket (This should fail).
    {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
```

```
        if (!listBucketsOutcome.IsSuccess()) {
            if (listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
            }
            else {
                std::cout
                        << "Access to list buckets denied because privileges have
not been applied."
                        << std::endl;
            }
        }
        else {
            std::cerr
                    << "Successfully retrieved bucket lists when this should not
happen."
                    << std::endl;
        }
    }

    // 6. Attach the policy to the role.
    {
        Aws::IAM::Model::AttachRolePolicyRequest request;
        request.SetRoleName(role.GetRoleName());
        request.WithPolicyArn(policy.GetArn());

        Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
                request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
                      ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
```

```
    }

    int count = 0;
    // 7. List objects in the bucket (this should succeed).
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
 role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
 s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
 LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
 std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }

            std::this_thread::sleep_for(std::chrono::seconds(1));
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
                      << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
    return DeleteCreatedEntities(client, role, user, policy);
}
```

```
bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
    bool result = true;
    if (policy.ArnHasBeenSet()) {
        // Detach the policy from the role.
        {
            Aws::IAM::Model::DetachRolePolicyRequest request;
            request.SetPolicyArn(policy.GetArn());
            request.SetRoleName(role.GetRoleName());

            Aws::IAM::Model::DetachRolePolicyOutcome outcome =
 client.DetachRolePolicy(
                    request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error Detaching policy from roles. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully detached the policy with arn "
                          << policy.GetArn()
                          << " from role " << role.GetRoleName() << "." <<
 std::endl;
            }
        }

        // Delete the policy.
        {
            Aws::IAM::Model::DeletePolicyRequest request;
            request.WithPolicyArn(policy.GetArn());

            Aws::IAM::Model::DeletePolicyOutcome outcome =
 client.DeletePolicy(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error deleting policy. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully deleted the policy with arn "
                          << policy.GetArn() << std::endl;
            }
```

```
        }

    }

    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour C++ .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Exécutez un scénario interactif à une invite de commande.

```
import (
 "context"
 "errors"
 "fmt"
 "log"
 "math/rand"
 "strings"
 "time"
```

```
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/credentials"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/aws-sdk-go-v2/service/s3"
 "github.com/aws/aws-sdk-go-v2/service/sts"
 "github.com/aws/smithy-go"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/iam/actions"
)

// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
//  1. Create a user who has no permissions.
//  2. Create a role that grants permission to list Amazon Simple Storage Service
//     (Amazon S3) buckets for the account.
//  3. Add a policy to let the user assume the role.
//  4. Try and fail to list buckets without permissions.
//  5. Assume the role and list S3 buckets using temporary credentials.
//  6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
 sdkConfig      aws.Config
 accountWrapper actions.AccountWrapper
 policyWrapper  actions.PolicyWrapper
 roleWrapper    actions.RoleWrapper
 userWrapper    actions.UserWrapper
 questioner     demotools.IQuestioner
 helper         IScenarioHelper
 isTestRun      bool
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
 helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
```

```
 return AssumeRoleScenario{
  sdkConfig:      sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
  questioner:     questioner,
  helper:         helper,
 }
}

// addTestOptions appends the API options specified in the original configuration
 to
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
}

// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run(ctx context.Context) {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong with the demo.\n")
   log.Println(r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
 demo.")
 log.Println(strings.Repeat("-", 88))

 user := scenario.CreateUser(ctx)
 accessKey := scenario.CreateAccessKey(ctx, user)
 role := scenario.CreateRoleAndPolicies(ctx, user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(ctx, accessKey)
 scenario.ListBucketsWithAssumedRole(ctx, noPermsConfig, role)
 scenario.Cleanup(ctx, user, role)
```

```
 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}

// CreateUser creates a new IAM user. This user has no permissions.
func (scenario AssumeRoleScenario) CreateUser(ctx context.Context) *types.User {
 log.Println("Let's create an example user with no permissions.")
 userName := scenario.questioner.Ask("Enter a name for the example user:",
 demotools.NotEmpty{})
 user, err := scenario.userWrapper.GetUser(ctx, userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(ctx, userName)
  if err != nil {
   panic(err)
  }
  log.Printf("Created user %v.\n", *user.UserName)
 } else {
  log.Printf("User %v already exists.\n", *user.UserName)
 }
 log.Println(strings.Repeat("-", 88))
 return user
}

// CreateAccessKey creates an access key for the user.
func (scenario AssumeRoleScenario) CreateAccessKey(ctx context.Context, user
 *types.User) *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(ctx, *user.UserName)
 if err != nil {
  panic(err)
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return accessKey
}

// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
```

```
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(ctx context.Context,
 user *types.User) *types.Role {
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err := scenario.roleWrapper.CreateRole(ctx,
 scenario.helper.GetName(), *user.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  ctx, scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"},
 "arn:aws:s3:::*")
 if err != nil {
  panic(err)
 }
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
 err = scenario.roleWrapper.AttachRolePolicy(ctx, *listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {
  panic(err)
 }
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
 err = scenario.userWrapper.CreateUserPolicy(ctx, *user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return listBucketsRole
}
```

```
// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(ctx
 context.Context, accessKey *types.AccessKey) *aws.Config {
 log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 noPermsConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
  ))
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)

 s3Client := s3.NewFromConfig(noPermsConfig)
 _, err = s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
    panic(err)
   }
  }
 } else {
  log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
   "but the call succeeded. Continuing the example anyway...")
```

```
 }
 log.Println(strings.Repeat("-", 88))
 return &noPermsConfig
}

// ListBucketsWithAssumedRole performs the following actions:
//
//  1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//     the user's access key credentials.
//  2. Gets temporary credentials by assuming the role that grants permission to
 list the
//     buckets.
//  3. Creates an Amazon S3 client from the temporary credentials.
//  4. Lists buckets for the account. Because the temporary credentials are
 generated by
//     assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(ctx
 context.Context, noPermsConfig *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(ctx, &sts.AssumeRoleInput{
  RoleArn:         role.Arn,
  RoleSessionName: aws.String("AssumeRoleExampleSession"),
  DurationSeconds: aws.Int32(900),
 })
 if err != nil {
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
   *tempCredentials.Credentials.SecretAccessKey,
   *tempCredentials.Credentials.SessionToken),
  ),
 )
 if err != nil {
  panic(err)
 }
```

```
 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&assumeRoleConfig)

 s3Client := s3.NewFromConfig(assumeRoleConfig)
 result, err := s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  log.Println("Couldn't list buckets with assumed role credentials.")
  panic(err)
 }
 log.Println("Successfully called ListBuckets with assumed role credentials, \n"
 +
  "here are some of them:")
 for i := 0; i < len(result.Buckets) && i < 5; i++ {
  log.Printf("\t%v\n", *result.Buckets[i].Name)
 }
 log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(ctx context.Context, user *types.User,
 role *types.Role) {
 if scenario.questioner.AskBool(
  "Do you want to delete the resources created for this example? (y/n)", "y",
 ) {
  policies, err := scenario.roleWrapper.ListAttachedRolePolicies(ctx,
 *role.RoleName)
  if err != nil {
   panic(err)
  }
  for _, policy := range policies {
   err = scenario.roleWrapper.DetachRolePolicy(ctx, *role.RoleName,
 *policy.PolicyArn)
   if err != nil {
    panic(err)
   }
   err = scenario.policyWrapper.DeletePolicy(ctx, *policy.PolicyArn)
   if err != nil {
    panic(err)
   }
   log.Printf("Detached policy %v from role %v and deleted the policy.\n",
    *policy.PolicyName, *role.RoleName)
  }
  err = scenario.roleWrapper.DeleteRole(ctx, *role.RoleName)
```

```
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted role %v.\n", *role.RoleName)

  userPols, err := scenario.userWrapper.ListUserPolicies(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  for _, userPol := range userPols {
   err = scenario.userWrapper.DeleteUserPolicy(ctx, *user.UserName, userPol)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
  }
  keys, err := scenario.userWrapper.ListAccessKeys(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  for _, key := range keys {
   err = scenario.userWrapper.DeleteAccessKey(ctx, *user.UserName,
 *key.AccessKeyId)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
 *user.UserName)
  }
  err = scenario.userWrapper.DeleteUser(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted user %v.\n", *user.UserName)
  log.Println(strings.Repeat("-", 88))
 }

}

// IScenarioHelper abstracts input and wait functions from a scenario so that
 they
// can be mocked for unit testing.
type IScenarioHelper interface {
 GetName() string
```

```
 Pause(secs int)
}

const rMax = 100000

type ScenarioHelper struct {
 Prefix string
 Random *rand.Rand
}

// GetName returns a unique name formed of a prefix and a random number.
func (helper *ScenarioHelper) GetName() string {
 return fmt.Sprintf("%v%v", helper.Prefix, helper.Random.Intn(rMax))
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
 time.Sleep(time.Duration(secs) * time.Second)
}
```

Définissez une structure qui encapsule les actions du compte.

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}
```

```
// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy(ctx context.Context)
 (*types.PasswordPolicy, error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(ctx,
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders(ctx context.Context)
 ([]types.SAMLProviderListEntry, error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(ctx,
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

Définissez une structure qui encapsule les actions de la politique.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
```

```
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(ctx context.Context, maxPolicies int32)
 ([]types.Policy, error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}



// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version   string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect    string
 Action    []string
 Principal map[string]string `json:",omitempty"`
 Resource  *string           `json:",omitempty"`
```

```
}

// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(ctx context.Context, policyName string,
 actions []string,
 resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(ctx, &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(ctx context.Context, policyArn string)
 (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(ctx, &iam.GetPolicyInput{
```

```
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(ctx context.Context, policyArn string)
 error {
 _, err := wrapper.IamClient.DeletePolicy(ctx, &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

Définissez une structure qui encapsule les actions du rôle.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
```

```
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(ctx context.Context, maxRoles int32)
 ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(ctx,
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(ctx context.Context, roleName string,
 trustedUserArn string) (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:    "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action:    []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
```

```
 }
 result, err := wrapper.IamClient.CreateRole(ctx, &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(ctx context.Context, roleName string)
 (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(ctx,
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(ctx context.Context,
 serviceName string, description string) (
 *types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(ctx,
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
```

```
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(ctx context.Context, roleName
 string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(ctx,
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(ctx context.Context, policyArn
 string, roleName string) error {
 _, err := wrapper.IamClient.AttachRolePolicy(ctx, &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
```

```
func (wrapper RoleWrapper) ListAttachedRolePolicies(ctx context.Context, roleName
 string) ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(ctx,
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(ctx context.Context, roleName string,
 policyArn string) error {
 _, err := wrapper.IamClient.DetachRolePolicy(ctx, &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(ctx context.Context, roleName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(ctx,
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
```

```
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(ctx context.Context, roleName string) error
 {
 _, err := wrapper.IamClient.DeleteRole(ctx, &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

Définissez une structure qui encapsule les actions de l'utilisateur.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
```

```
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(ctx context.Context, maxUsers int32)
 ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(ctx, &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(ctx, &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
```

```
 return user, err
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(ctx, &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(ctx context.Context, userName string,
 policyName string, actions []string,
 roleArn string) error {
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
```

```
 }
 _, err = wrapper.IamClient.PutUserPolicy(ctx, &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(ctx context.Context, userName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(ctx,
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(ctx context.Context, userName string,
 policyName string) error {
 _, err := wrapper.IamClient.DeleteUserPolicy(ctx, &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
```

```
 }
 return err
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(ctx context.Context, userName string) error
 {
 _, err := wrapper.IamClient.DeleteUser(ctx, &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(ctx context.Context, userName
 string) (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(ctx, &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(ctx context.Context, userName string,
 keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(ctx, &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
```

```
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(ctx context.Context, userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(ctx, &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour Go .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser

- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de l'utilisateur IAM.

```
/*
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
 */

public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
```

```
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

   public static String userArn;

   public static void main(String[] args) throws Exception {

       final String usage = """

               Usage:
                   <username> <policyName> <roleName> <roleSessionName>
<bucketName>\s

               Where:
                   username - The name of the IAM user to create.\s
                   policyName - The name of the policy to create.\s
                   roleName - The name of the role to create.\s
                   roleSessionName - The name of the session required for the
assumeRole operation.\s
                   bucketName - The name of the Amazon S3 bucket from which
objects are read.\s
               """;

       if (args.length != 5) {
           System.out.println(usage);
           System.exit(1);
       }

       String userName = args[0];
       String policyName = args[1];
       String roleName = args[2];
       String roleSessionName = args[3];
       String bucketName = args[4];

       Region region = Region.AWS_GLOBAL;
       IamClient iam = IamClient.builder()
```

```
                .region(region)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the AWS IAM example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
```

```
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
```

```
        }
        return "";
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);

waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();

            ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
            String polArn;
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
```

```
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
this role.");
                    return;
                }
            }

            AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);
            System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();

        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();
```

```
            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();

            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();

            System.out.println("Created a S3Client using temp credentials.");
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void deleteRole(IamClient iam, String roleName, String polArn)
{

        try {
            // First the policy needs to be detached.
            DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
```

```
                    .policyArn(polArn)
                    .roleName(roleName)
                    .build();

            iam.detachRolePolicy(rolePolicyRequest);

            // Delete the policy.
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(polArn)
                    .build();

            iam.deletePolicy(request);
            System.out.println("*** Successfully deleted " + polArn);

            // Delete the role.
            DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();

            iam.deleteRole(roleRequest);
            System.out.println("*** Successfully deleted " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteKey(IamClient iam, String username, String
accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
```

```
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK for Java 2.x .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créer un utilisateur IAM et un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3. L'utilisateur n'a que le droit d'assumer le rôle. Après avoir assumé le rôle, utilisez des informations d'identification temporaires pour répertorier les compartiments pour le compte.

```
import {
  CreateUserCommand,
  GetUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import { ScenarioInput } from "@aws-doc-sdk-examples/lib/scenario/index.js";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "iam_basic_test_username";
const policyName = "iam_basic_test_policy";
const roleName = "iam_basic_test_role";

/**
 * Create a new IAM user. If the user already exists, give
```

```
 * the option to delete and re-create it.
 * @param {string} name
 */
export const createUser = async (name, confirmAll = false) => {
  try {
    const { User } = await iamClient.send(
      new GetUserCommand({ UserName: name }),
    );
    const input = new ScenarioInput(
      "deleteUser",
      "Do you want to delete and remake this user?",
      { type: "confirm" },
    );
    const deleteUser = await input.handle({}, { confirmAll });
    // If the user exists, and you want to delete it, delete the user
    // and then create it again.
    if (deleteUser) {
      await iamClient.send(new DeleteUserCommand({ UserName: User.UserName }));
      await iamClient.send(new CreateUserCommand({ UserName: name }));
    } else {
      console.warn(
        `${name} already exists. The scenario may not work as expected.`,
      );
      return User;
    }
  } catch (caught) {
    // If there is no user by that name, create one.
    if (caught instanceof Error && caught.name === "NoSuchEntityException") {
      const { User } = await iamClient.send(
        new CreateUserCommand({ UserName: name }),
      );
      return User;
    }
    throw caught;
  }
};

export const main = async (confirmAll = false) => {
  // Create a user. The user has no permissions by default.
  const User = await createUser(userName, confirmAll);

  if (!User) {
    throw new Error("User not created");
  }
```

```
 // Create an access key. This key is used to authenticate the new user to
 // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
(AWS STS).
 // It's not best practice to use access keys. For more information, see
https://aws.amazon.com/iam/resources/best-practices/.
 const createAccessKeyResponse = await iamClient.send(
   new CreateAccessKeyCommand({ UserName: userName }),
 );

 if (
   !createAccessKeyResponse.AccessKey?.AccessKeyId ||
   !createAccessKeyResponse.AccessKey?.SecretAccessKey
 ) {
   throw new Error("Access key not created");
 }

 const {
   AccessKey: { AccessKeyId, SecretAccessKey },
 } = createAccessKeyResponse;

 let s3Client = new S3Client({
   credentials: {
     accessKeyId: AccessKeyId,
     secretAccessKey: SecretAccessKey,
   },
 });

 // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
 // thrown while the user and access keys are still stabilizing.
 await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
   try {
     return await listBuckets(s3Client);
   } catch (err) {
     if (err instanceof Error && err.name === "InvalidAccessKeyId") {
       throw err;
     }
   }
 });

 // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
 // is thrown while the user and access keys are still stabilizing.
 const { Role } = await retry(
```

```
    {
      intervalInMs: 2000,
      maxRetries: 60,
    },
    () =>
      iamClient.send(
        new CreateRoleCommand({
          AssumeRolePolicyDocument: JSON.stringify({
            Version: "2012-10-17",
            Statement: [
              {
                Effect: "Allow",
                Principal: {
                  // Allow the previously created user to assume this role.
                  AWS: User.Arn,
                },
                Action: "sts:AssumeRole",
              },
            ],
          }),
          RoleName: roleName,
        }),
      ),
  );

  if (!Role) {
    throw new Error("Role not created");
  }

  // Create a policy that allows the user to list S3 buckets.
  const { Policy: listBucketPolicy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
            Effect: "Allow",
            Action: ["s3:ListAllMyBuckets"],
            Resource: "*",
          },
        ],
      }),
      PolicyName: policyName,
    }),
```

```
);

if (!listBucketPolicy) {
  throw new Error("Policy not created");
}

// Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
await iamClient.send(
  new AttachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
);

// Assume the role.
const stsClient = new STSClient({
  credentials: {
    accessKeyId: AccessKeyId,
    secretAccessKey: SecretAccessKey,
  },
});

// Retry the assume role operation until it succeeds.
const { Credentials } = await retry(
  { intervalInMs: 2000, maxRetries: 60 },
  () =>
    stsClient.send(
      new AssumeRoleCommand({
        RoleArn: Role.Arn,
        RoleSessionName: `iamBasicScenarioSession-${Math.floor(
          Math.random() * 1000000,
        )}`,
        DurationSeconds: 900,
      }),
    ),
);

if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
  throw new Error("Credentials not created");
}

s3Client = new S3Client({
  credentials: {
    accessKeyId: Credentials.AccessKeyId,
```

```
    secretAccessKey: Credentials.SecretAccessKey,
    sessionToken: Credentials.SessionToken,
  },
});

// List the S3 buckets again.
// Retry the list buckets operation until it succeeds. AccessDenied might
// be thrown while the role policy is still stabilizing.
await retry({ intervalInMs: 2000, maxRetries: 120 }, () =>
  listBuckets(s3Client),
);

// Clean up.
await iamClient.send(
  new DetachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
);

await iamClient.send(
  new DeletePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
  }),
);

await iamClient.send(
  new DeleteRoleCommand({
    RoleName: Role.RoleName,
  }),
);

await iamClient.send(
  new DeleteAccessKeyCommand({
    UserName: userName,
    AccessKeyId,
  }),
);

await iamClient.send(
  new DeleteUserCommand({
    UserName: userName,
  }),
);
```

```
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
    throw new Error("Buckets not listed");
  }

  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour JavaScript .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de l'utilisateur IAM.

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <username> <policyName> <roleName> <roleSessionName> <fileLocation>
 <bucketName>

    Where:
        username - The name of the IAM user to create.
        policyName - The name of the policy to create.
        roleName - The name of the role to create.
        roleSessionName - The name of the session required for the assumeRole
 operation.
        fileLocation - The file location to the JSON required to create the role
 (see Readme).
        bucketName - The name of the Amazon S3 bucket from which objects are
 read.
    """

    if (args.size != 6) {
        println(usage)
        exitProcess(1)
    }

    val userName = args[0]
    val policyName = args[1]
    val roleName = args[2]
    val roleSessionName = args[3]
    val fileLocation = args[4]
    val bucketName = args[5]

    createUser(userName)
```

```
    println("$userName was successfully created.")

    val polArn = createPolicy(policyName)
    println("The policy $polArn was successfully created.")

    val roleArn = createRole(roleName, fileLocation)
    println("$roleArn was successfully created.")
    attachRolePolicy(roleName, polArn)

    println("*** Wait for 1 MIN so the resource is available.")
    delay(60000)
    assumeGivenRole(roleArn, roleSessionName, bucketName)

    println("*** Getting ready to delete the AWS resources.")
    deleteRole(roleName, polArn)
    deleteUser(userName)
    println("This IAM Scenario has successfully completed.")
}

suspend fun createUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {
    val policyDocumentValue = """
    {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "s3:*"
                ],
                "Resource": "*"
            }
        ]
```

```
    }
    """.trimIndent()

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentValue
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}

suspend fun createRole(
    rolenameVal: String?,
    fileLocation: String?,
): String? {
    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request =
        CreateRoleRequest {
            roleName = rolenameVal
            assumeRolePolicyDocument = jsonObject.toJSONString()
            description = "Created using the AWS SDK for Kotlin"
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
```

```
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkMyList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String,
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(
    roleArnVal: String?,
    roleSessionNameVal: String?,
    bucketName: String,
) {
    val stsClient = StsClient.fromEnvironment { region = "us-east-1" }
```

```
    val roleRequest =
        AssumeRoleRequest {
            roleArn = roleArnVal
            roleSessionName = roleSessionNameVal
        }

    val roleResponse = stsClient.assumeRole(roleRequest)
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken

    val staticCredentials = StaticCredentialsProvider {
        accessKeyId = key
        secretAccessKey = secKey
        sessionToken = secToken
    }

    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 = S3Client.fromEnvironment {
        region = "us-east-1"
        credentialsProvider = staticCredentials
    }

    println("Created a S3Client using temp credentials.")
    println("Listing objects in $bucketName")

    val listObjects =
        ListObjectsRequest {
            bucket = bucketName
        }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
    }
}

suspend fun deleteRole(
    roleNameVal: String,
    polArn: String,
) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }
```

```
    // First the policy needs to be detached.
    val rolePolicyRequest =
        DetachRolePolicyRequest {
            policyArn = polArn
            roleName = roleNameVal
        }

    iam.detachRolePolicy(rolePolicyRequest)

    // Delete the policy.
    val request =
        DeletePolicyRequest {
            policyArn = polArn
        }

    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")

    // Delete the role.
    val roleRequest =
        DeleteRoleRequest {
            roleName = roleNameVal
        }

    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}

@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
```

```
    return jsonParser.parse(reader)
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour Kotlin.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
namespace Iam\Basics;

require 'vendor/autoload.php';

use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
```

```
use Aws\Sts\StsClient;
use Iam\IAMService;

echo("\n");
echo("---------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("---------------------------------------\n");

$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";

$key = $service->createAccessKey($user['UserName']);
$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
```

```
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
    ]);
    echo "this should not run";
} catch (S3Exception $exception) {
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
    echo "this should now not fail\n";
}

$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
```

```
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour PHP .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créer un utilisateur IAM et un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3. L'utilisateur n'a que le droit d'assumer le rôle. Après avoir assumé le rôle, utilisez des informations d'identification temporaires pour répertorier les compartiments pour le compte.

```
import json
import sys
import time
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError


def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
        time.sleep(1)
        print(".", end="")
        sys.stdout.flush()
    print()


def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, roles, and
 policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise
```

```
    try:
        user_key = user.create_access_key_pair()
        print(f"Created access key pair for user.")
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    try:
        role = iam_resource.create_role(
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
                    ],
                }
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        policy = iam_resource.create_policy(
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
```

```
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "s3:ListAllMyBuckets",
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
    except ClientError as error:
        print(
            f"Couldn't create a policy and attach it to role {role.name}. Here's
why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user.create_policy(
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
            f"Created an inline policy for {user.name} that lets the user assume
"
            f"the role."
        )
    except ClientError as error:
        print(
```

```
            f"Couldn't create an inline policy for user {user.name}. Here's why:
 "
            f"{error.response['Error']['Message']}"
        )
        raise

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, role


def show_access_denied_without_role(user_key):
    """
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
        "s3", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
        raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Attempt to list buckets with no permissions: AccessDenied.")
        else:
            raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the
 account.
    Uses the temporary credentials from the role to list the buckets that are
 owned
    by the assumed role's account.
```

```
    :param user_key: The access key of a user that has permission to assume the
role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id,
aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    # Create an S3 resource that can access the account with the temporary
credentials.
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
            f"Couldn't list buckets for the account. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise
```

```
def teardown(user, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        for user_pol in user.policies.all():
            user_pol.delete()
            print("Deleted inline user policy.")
        for key in user.access_keys.all():
            key.delete()
            print("Deleted user's access key.")
        user.delete()
        print(f"Deleted {user.name}.")
    except ClientError as error:
        print(
            "Couldn't delete user policy or delete user. Here's why: "
            f"{error.response['Error']['Message']}"
        )


def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
```

```
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
    role = None
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
        if user is not None and role is not None:
            teardown(user, role)
        print("Thanks for watching!")


if __name__ == "__main__":
    usage_demo()
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK for Python (Boto3).
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créer un utilisateur IAM et un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3. L'utilisateur n'a que le droit d'assumer le rôle. Après avoir assumé le rôle, utilisez des informations d'identification temporaires pour répertorier les compartiments pour le compte.

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client

  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Waits for the specified number of seconds.
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts('Give AWS time to propagate resources...')
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info('Tried and failed to create demo user.')
```

```
    @logger.info("\t#{e.code}: #{e.message}")
    @logger.info("\nCan't continue the demo without a user!")
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
  # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
  # @return [Aws::IAM::Role] The newly created role.
  def create_role(role_name, user)
    trust_policy = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Principal: { 'AWS': user.arn },
        Action: 'sts:AssumeRole'
      }]
    }.to_json
    role = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: trust_policy
    ).role
    @logger.info("Created role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
```

```
    @logger.info("Couldn't create a role for the demo. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end

  # Creates a policy that grants permission to list S3 buckets in the account,
and
  # then attaches the policy to a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param role [Aws::IAM::Role] The role that the policy is attached to.
  # @return [Aws::IAM::Policy] The newly created policy.
  def create_and_attach_role_policy(policy_name, role)
    policy_document = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Action: 's3:ListAllMyBuckets',
        Resource: 'arn:aws:s3:::*'
      }]
    }.to_json
    policy = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document
    ).policy
    @iam_client.attach_role_policy(
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an inline policy for a user that lets the user assume a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param user [Aws::IAM::User] The user that owns the policy.
```

```
 # @param role [Aws::IAM::Role] The role that can be assumed.
 # @return [Aws::IAM::UserPolicy] The newly created policy.
 def create_user_policy(policy_name, user, role)
   policy_document = {
     Version: '2012-10-17',
     Statement: [{
       Effect: 'Allow',
       Action: 'sts:AssumeRole',
       Resource: role.arn
     }]
   }.to_json
   @iam_client.put_user_policy(
     user_name: user.user_name,
     policy_name: policy_name,
     policy_document: policy_document
   )
   puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
 rescue Aws::Errors::ServiceError => e
   @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
   @logger.info("\t#{e.code}: #{e.message}")
   raise
 end

 # Creates an Amazon S3 resource with specified credentials. This is separated
into a
 # factory function so that it can be mocked for unit testing.
 #
 # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
 def create_s3_resource(credentials)
   Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
 end

 # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
 # Because the resource uses credentials with limited access, it may not be able
to
 # list the S3 buckets.
 #
 # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
 def list_buckets(s3_resource)
   count = 10
   s3_resource.buckets.each do |bucket|
```

```
      @logger.info "\t#{bucket.name}"
      count -= 1
      break if count.zero?
    end
  rescue Aws::Errors::ServiceError => e
    if e.code == 'AccessDenied'
      puts('Attempt to list buckets with no permissions: AccessDenied.')
    else
      @logger.info("Couldn't list buckets for the account. Here's why: ")
      @logger.info("\t#{e.code}: #{e.message}")
      raise
    end
  end

  # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
  # This is separated into a factory function so that it can be mocked for unit
testing.
  #
  # @param key_id [String] The ID of the access key used by the STS client.
  # @param key_secret [String] The secret part of the access key used by the STS
client.
  def create_sts_client(key_id, key_secret)
    Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
  end

  # Gets temporary credentials that can be used to assume a role.
  #
  # @param role_arn [String] The ARN of the role that is assumed when these
credentials
  #                          are used.
  # @param sts_client [AWS::STS::Client] An AWS STS client.
  # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
  def assume_role(role_arn, sts_client)
    credentials = Aws::AssumeRoleCredentials.new(
      client: sts_client,
      role_arn: role_arn,
      role_session_name: 'create-use-assume-role-scenario'
    )
    @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
    credentials
  end
```

```
  # Deletes a role. If the role has policies attached, they are detached and
  # deleted before the role is deleted.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    @iam_client.list_attached_role_policies(role_name:
 role_name).attached_policies.each do |policy|
      @iam_client.detach_role_policy(role_name: role_name, policy_arn:
 policy.policy_arn)
      @iam_client.delete_policy(policy_arn: policy.policy_arn)
      @logger.info("Detached and deleted policy #{policy.policy_name}.")
    end
    @iam_client.delete_role({ role_name: role_name })
    @logger.info("Role deleted: #{role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
 why:")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Deletes a user. If the user has inline policies or access keys, they are
 deleted
  # before the user is deleted.
  #
  # @param user [Aws::IAM::User] The user to delete.
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
```

```
  puts('-' * 88)
  puts('Welcome to the IAM create a user and assume a role demo!')
  puts('-' * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts('Try to list buckets with credentials for a user who has no permissions.')
  puts('Expect AccessDenied from this call.')
  scenario.list_buckets(
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key))
  )
  puts('Now, assume the role that grants permission.')
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key)
  )
  puts('Here are your buckets:')
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
  scenario.delete_user(user.user_name)
  puts('Thanks for watching!')
  puts('-' * 88)
rescue Aws::Errors::ServiceError => e
  puts('Something went wrong with the demo.')
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour Ruby .
  - AttachRolePolicy

- CreateAccessKey
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
        initialize_variables().await;
```

```
    if let Err(e) = run_iam_operations(
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (iamClient, String, String, String) {
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();

    (
        client,
        uuid,
```

```
        list_all_buckets_policy_document,
        inline_policy_document,
    )
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
    let key = iam_service::create_access_key(&client, user.user_name()).await?;

    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
    println!(
        "Created policy: {}",
        list_all_buckets_policy.policy_name.as_ref().unwrap()
```

```
    );

    let attach_role_policy_result =
        iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
            .await?;
    println!(
        "Attached the policy to the role: {:?}",
        attach_role_policy_result
    );

    let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
    let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
    iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
        .await?;
    println!("Created inline policy.");

    //First, fail to list the buckets with the user.
    let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
    let fail_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    println!("Fail config: {:?}", fail_config);
    let fail_client: s3Client = s3Client::new(&fail_config);
    match fail_client.list_buckets().send().await {
        Ok(e) => {
            println!("This should not run. {:?}", e);
        }
        Err(e) => {
            println!("Successfully failed with error: {:?}", e)
        }
    }

    let sts_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    let sts_client: stsClient = stsClient::new(&sts_config);
    sleep(Duration::from_secs(10)).await;
    let assumed_role = sts_client
```

```
        .assume_role()
        .role_arn(assume_role_role.arn())
        .role_session_name(format!("iam_demo_assumerole_session_{uuid}"))
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .access_key_id(),
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
                .as_ref()
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
```

```
            println!("This should now run successfully.")
        }
        Err(e) => {
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
    .await?;
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour Rust.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole

- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Actions destinées à l'utilisation d'IAM AWS Kits SDK

Les exemples de code suivants montrent comment effectuer des actions IAM individuelles avec des AWS SDK. Chaque exemple inclut un lien vers GitHub, où vous pouvez trouver des instructions pour configurer et exécuter le code.

Ces extraits appellent l'API IAM et sont des extraits de code de programmes plus volumineux qui doivent être exécutés en contexte. Vous pouvez voir les actions dans leur contexte dans Scénarios d'utilisation d'IAM AWS Kits SDK.

Les exemples suivants incluent uniquement les actions les plus couramment utilisées. Pour obtenir la liste complète, consultez la Référence des API Gestion des identités et des accès AWS.

Exemples

- Utilisation de AddClientIdToOpenIdConnectProvider avec une CLI
- Utilisation de AddRoleToInstanceProfile avec une CLI
- Utilisation de AddUserToGroup avec une CLI
- Utilisation de AttachGroupPolicy avec une CLI
- À utiliser AttachRolePolicyavec un AWS SDK ou CLI
- À utiliser AttachUserPolicyavec un AWS SDK ou CLI
- Utilisation de ChangePassword avec une CLI
- À utiliser CreateAccessKeyavec un AWS SDK ou CLI
- À utiliser CreateAccountAliasavec un AWS SDK ou CLI
- À utiliser CreateGroupavec un AWS SDK ou CLI
- À utiliser CreateInstanceProfileavec un AWS SDK ou CLI

- Utilisation de CreateLoginProfile avec une CLI
- Utilisation de CreateOpenIdConnectProvider avec une CLI
- À utiliser CreatePolicyavec un AWS SDK ou CLI
- À utiliser CreatePolicyVersionavec un AWS SDK ou CLI
- À utiliser CreateRoleavec un AWS SDK ou CLI
- Utilisez CreateSAMLProvider avec un AWS SDK ou CLI
- À utiliser CreateServiceLinkedRoleavec un AWS SDK ou CLI
- À utiliser CreateUseravec un AWS SDK ou CLI
- Utilisation de CreateVirtualMfaDevice avec une CLI
- Utilisation de DeactivateMfaDevice avec une CLI
- À utiliser DeleteAccessKeyavec un AWS SDK ou CLI
- À utiliser DeleteAccountAliasavec un AWS SDK ou CLI
- Utilisation de DeleteAccountPasswordPolicy avec une CLI
- À utiliser DeleteGroupavec un AWS SDK ou CLI
- Utilisation de DeleteGroupPolicy avec une CLI
- À utiliser DeleteInstanceProfileavec un AWS SDK ou CLI
- Utilisation de DeleteLoginProfile avec une CLI
- Utilisation de DeleteOpenIdConnectProvider avec une CLI
- À utiliser DeletePolicyavec un AWS SDK ou CLI
- À utiliser DeletePolicyVersionavec un AWS SDK ou CLI
- À utiliser DeleteRoleavec un AWS SDK ou CLI
- Utilisation de DeleteRolePermissionsBoundary avec une CLI
- À utiliser DeleteRolePolicyavec un AWS SDK ou CLI
- Utilisez DeleteSAMLProvider avec un AWS SDK ou CLI
- À utiliser DeleteServerCertificateavec un AWS SDK ou CLI
- À utiliser DeleteServiceLinkedRoleavec un AWS SDK ou CLI
- Utilisation de DeleteSigningCertificate avec une CLI
- À utiliser DeleteUseravec un AWS SDK ou CLI

- Utilisation de DeleteUserPermissionsBoundary avec une CLI
- À utiliser DeleteUserPolicyavec un AWS SDK ou CLI
- Utilisation de DeleteVirtualMfaDevice avec une CLI
- Utilisation de DetachGroupPolicy avec une CLI
- À utiliser DetachRolePolicyavec un AWS SDK ou CLI
- À utiliser DetachUserPolicyavec un AWS SDK ou CLI
- Utilisation de EnableMfaDevice avec une CLI
- À utiliser GenerateCredentialReportavec un AWS SDK ou CLI
- Utilisation de GenerateServiceLastAccessedDetails avec une CLI
- À utiliser GetAccessKeyLastUsedavec un AWS SDK ou CLI
- À utiliser GetAccountAuthorizationDetailsavec un AWS SDK ou CLI
- À utiliser GetAccountPasswordPolicyavec un AWS SDK ou CLI
- À utiliser GetAccountSummaryavec un AWS SDK ou CLI
- Utilisation de GetContextKeysForCustomPolicy avec une CLI
- Utilisation de GetContextKeysForPrincipalPolicy avec une CLI
- À utiliser GetCredentialReportavec un AWS SDK ou CLI
- Utilisation de GetGroup avec une CLI
- Utilisation de GetGroupPolicy avec une CLI
- Utilisation de GetInstanceProfile avec une CLI
- Utilisation de GetLoginProfile avec une CLI
- Utilisation de GetOpenIdConnectProvider avec une CLI
- À utiliser GetPolicyavec un AWS SDK ou CLI
- À utiliser GetPolicyVersionavec un AWS SDK ou CLI
- À utiliser GetRoleavec un AWS SDK ou CLI
- Utilisation de GetRolePolicy avec une CLI
- Utilisation de GetSamlProvider avec une CLI
- À utiliser GetServerCertificateavec un AWS SDK ou CLI
- Utilisation de GetServiceLastAccessedDetails avec une CLI

- Utilisation de GetServiceLastAccessedDetailsWithEntities avec une CLI
- À utiliser GetServiceLinkedRoleDeletionStatusavec un AWS SDK ou CLI
- À utiliser GetUseravec un AWS SDK ou CLI
- Utilisation de GetUserPolicy avec une CLI
- À utiliser ListAccessKeysavec un AWS SDK ou CLI
- À utiliser ListAccountAliasesavec un AWS SDK ou CLI
- Utilisation de ListAttachedGroupPolicies avec une CLI
- À utiliser ListAttachedRolePoliciesavec un AWS SDK ou CLI
- Utilisation de ListAttachedUserPolicies avec une CLI
- Utilisation de ListEntitiesForPolicy avec une CLI
- Utilisation de ListGroupPolicies avec une CLI
- À utiliser ListGroupsavec un AWS SDK ou CLI
- Utilisation de ListGroupsForUser avec une CLI
- Utilisation de ListInstanceProfiles avec une CLI
- Utilisation de ListInstanceProfilesForRole avec une CLI
- Utilisation de ListMfaDevices avec une CLI
- Utilisation de ListOpenIdConnectProviders avec une CLI
- À utiliser ListPoliciesavec un AWS SDK ou CLI
- À utiliser ListPolicyVersionsavec un AWS SDK ou CLI
- À utiliser ListRolePoliciesavec un AWS SDK ou CLI
- Utilisation de ListRoleTags avec une CLI
- À utiliser ListRolesavec un AWS SDK ou CLI
- Utilisez ListSamlProviders avec un AWS SDK ou CLI
- À utiliser ListServerCertificatesavec un AWS SDK ou CLI
- Utilisation de ListSigningCertificates avec une CLI
- À utiliser ListUserPoliciesavec un AWS SDK ou CLI
- Utilisation de ListUserTags avec une CLI
- À utiliser ListUsersavec un AWS SDK ou CLI

- Utilisation de ListVirtualMfaDevices avec une CLI
- Utilisation de PutGroupPolicy avec une CLI
- Utilisation de PutRolePermissionsBoundary avec une CLI
- À utiliser PutRolePolicyavec un AWS SDK ou CLI
- Utilisation de PutUserPermissionsBoundary avec une CLI
- À utiliser PutUserPolicyavec un AWS SDK ou CLI
- Utilisation de RemoveClientIdFromOpenIdConnectProvider avec une CLI
- Utilisation de RemoveRoleFromInstanceProfile avec une CLI
- Utilisation de RemoveUserFromGroup avec une CLI
- Utilisation de ResyncMfaDevice avec une CLI
- À utiliser SetDefaultPolicyVersionavec un AWS SDK ou CLI
- Utilisation de TagRole avec une CLI
- Utilisation de TagUser avec une CLI
- Utilisation de UntagRole avec une CLI
- Utilisation de UntagUser avec une CLI
- À utiliser UpdateAccessKeyavec un AWS SDK ou CLI
- Utilisation de UpdateAccountPasswordPolicy avec une CLI
- Utilisation de UpdateAssumeRolePolicy avec une CLI
- Utilisation de UpdateGroup avec une CLI
- Utilisation de UpdateLoginProfile avec une CLI
- Utilisation de UpdateOpenIdConnectProviderThumbprint avec une CLI
- Utilisation de UpdateRole avec une CLI
- Utilisation de UpdateRoleDescription avec une CLI
- Utilisation de UpdateSamlProvider avec une CLI
- À utiliser UpdateServerCertificateavec un AWS SDK ou CLI
- Utilisation de UpdateSigningCertificate avec une CLI
- À utiliser UpdateUseravec un AWS SDK ou CLI
- À utiliser UploadServerCertificateavec un AWS SDK ou CLI
- Utilisation de UploadSigningCertificate avec une CLI

Utilisation de **`AddClientIdToOpenIdConnectProvider`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `AddClientIdToOpenIdConnectProvider`.

CLI

AWS CLI

Pour ajouter un identifiant client (audience) à un fournisseur Open-ID Connect (OIDC)

La commande `add-client-id-to-open-id-connect-provider` suivante ajoute l'ID client `my-application-ID` au fournisseur OIDC nommé `server.example.com`.

```
aws iam add-client-id-to-open-id-connect-provider \
    --client-id my-application-ID \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

Cette commande ne produit aucune sortie.

Pour créer un fournisseur OIDC, utilisez la commande `create-open-id-connect-provider`.

Pour plus d'informations, consultez Création de fournisseurs d'identité OpenID Connect (OIDC) du Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous AddClientIdToOpenIdConnectProviderà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cette commande ajoute l'ID client (ou audience) **`my-application-ID`** au fournisseur OIDC existant nommé **`server.example.com`**.

```
Add-IAMClientIDToOpenIDConnectProvider -ClientID "my-application-ID"
 -OpenIDConnectProviderARN "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
```

- Pour plus de détails sur l'API, reportez-vous AddClientIdToOpenIdConnectProviderà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cette commande ajoute l'ID client (ou audience) **`my-application-ID`** au fournisseur OIDC existant nommé **`server.example.com`**.

```
Add-IAMClientIDToOpenIDConnectProvider -ClientID "my-application-ID"
 -OpenIDConnectProviderARN "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
```

- Pour plus de détails sur l'API, reportez-vous AddClientIdToOpenIdConnectProviderà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`AddRoleToInstanceProfile`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `AddRoleToInstanceProfile`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Création d'un service de conteneur pour les instances de machines virtuelles
- Débuter avec le streaming géré
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts

CLI

AWS CLI

Pour ajouter un rôle à un profil d'instance

La commande `add-role-to-instance-profile` suivante ajoute le rôle nommé `S3Access` au profil d'instance nommé `Webserver`.

```
aws iam add-role-to-instance-profile \
    --role-name S3Access \
    --instance-profile-name Webserver
```

Cette commande ne produit aucune sortie.

Pour créer un profil d'instance, utilisez la commande `create-instance-profile`.

Pour plus d'informations, consultez Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2 dans le AWS Guide de l'utilisateur IAM.

- Pour plus de détails sur l'API, reportez-vous AddRoleToInstanceProfileà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cette commande ajoute le rôle nommé **S3Access** à un profil d'instance existant nommé **webserver**. Pour créer le profil d'instance, utilisez la commande **New-IAMInstanceProfile**. Après avoir créé le profil d'instance et l'avoir associé à un rôle à l'aide de cette commande, vous pouvez l'associer à une instance EC2. Pour ce faire, utilisez l'applet de commande **New-EC2Instance** avec le paramètre **InstanceProfile_Arn** ou **InstanceProfile-Name** pour lancer la nouvelle instance.

```
Add-IAMRoleToInstanceProfile -RoleName "S3Access" -InstanceProfileName
 "webserver"
```

- Pour plus de détails sur l'API, reportez-vous AddRoleToInstanceProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cette commande ajoute le rôle nommé **S3Access** à un profil d'instance existant nommé **webserver**. Pour créer le profil d'instance, utilisez la commande **New-IAMInstanceProfile**. Après avoir créé le profil d'instance et l'avoir associé à un rôle à l'aide de cette commande, vous pouvez l'associer à une instance EC2. Pour ce faire, utilisez l'applet de commande **New-EC2Instance** avec le paramètre **InstanceProfile_Arn** ou **InstanceProfile-Name** pour lancer la nouvelle instance.

```
Add-IAMRoleToInstanceProfile -RoleName "S3Access" -InstanceProfileName
 "webserver"
```

- Pour plus de détails sur l'API, reportez-vous AddRoleToInstanceProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`AddUserToGroup`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `AddUserToGroup`.

CLI

AWS CLI

Pour ajouter un utilisateur à un groupe IAM

La commande `add-user-to-group` suivante ajoute un utilisateur IAM nommé `Bob` au groupe IAM nommé `Admins`.

```
aws iam add-user-to-group \
    --user-name Bob \
    --group-name Admins
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Ajout et suppression d'utilisateurs dans un groupe d'utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous AddUserToGroupà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cette commande ajoute l'utilisateur nommé **`Bob`** au groupe nommé **`Admins`**.

```
Add-IAMUserToGroup -UserName "Bob" -GroupName "Admins"
```

- Pour plus de détails sur l'API, reportez-vous AddUserToGroupà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cette commande ajoute l'utilisateur nommé **Bob** au groupe nommé **Admins**.

```
Add-IAMUserToGroup -UserName "Bob" -GroupName "Admins"
```

- Pour plus de détails sur l'API, reportez-vous AddUserToGroupà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`AttachGroupPolicy`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `AttachGroupPolicy`.

CLI

AWS CLI

Pour attacher une politique gérée à un groupe IAM

La `attach-group-policy` commande suivante associe la politique AWS gérée nommée `ReadOnlyAccess` au groupe IAM nomméFinance.

```
aws iam attach-group-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --group-name Finance
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Politiques gérées et politiques en ligne dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous AttachGroupPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple associe la politique gérée par le client, nommée **`TesterPolicy`**, au groupe IAM **`Testers`**. Les utilisateurs de ce groupe sont immédiatement affectés par les autorisations définies dans la version par défaut de cette politique.

```
Register-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Exemple 2 : Cet exemple associe la politique AWS gérée nommée **`AdministratorAccess`** au groupe **`Admins`** IAM. Les utilisateurs de ce groupe sont immédiatement affectés par les autorisations définies dans la version la plus récente de cette politique.

```
Register-IAMGroupPolicy -GroupName Admins -PolicyArn arn:aws:iam::aws:policy/
AdministratorAccess
```

- Pour plus de détails sur l'API, reportez-vous AttachGroupPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple associe la politique gérée par le client, nommée **`TesterPolicy`**, au groupe IAM **`Testers`**. Les utilisateurs de ce groupe sont immédiatement affectés par les autorisations définies dans la version par défaut de cette politique.

```
Register-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Exemple 2 : Cet exemple associe la politique AWS gérée nommée **`AdministratorAccess`** au groupe **`Admins`** IAM. Les utilisateurs de ce groupe sont immédiatement affectés par les autorisations définies dans la version la plus récente de cette politique.

```
Register-IAMGroupPolicy -GroupName Admins -PolicyArn arn:aws:iam::aws:policy/
AdministratorAccess
```

- Pour plus de détails sur l'API, reportez-vous AttachGroupPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`AttachRolePolicy`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `AttachRolePolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Principes de base
- Configuration de la connectivité des services de conteneurs
- Création d'une tâche de conteneur pour le type de lancement sans serveur
- Création d'une API REST avec intégration de proxy de fonctions
- Création d'un service de conteneur pour les instances de machines virtuelles
- Création d'un espace de travail de surveillance géré
- Création d'un tableau de bord de surveillance avec le nom de la fonction comme variable
- Création de votre première fonction sans serveur
- Commencer à gérer la configuration
- Débuter avec la protection des appareils liés à l'Internet des objets
- Commencer à utiliser les magasins de fonctionnalités d'apprentissage automatique
- Commencer à utiliser les clusters Kubernetes gérés
- Débuter avec le streaming géré
- Commencer à orchestrer les flux de travail
- Gérer les rôles
- Transférer les secrets codés en dur vers un stockage secret sécurisé
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts
- Configuration de la gestion des systèmes
- Utilisation de variables de propriété dans les tableaux de bord de surveillance pour surveiller plusieurs fonctions sans serveur
- Travaillez avec Streams et Time-to-Live

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous AttachRolePolicyà la section Référence des AWS SDK pour .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
```

```
  echo "  -p policy_ARN -- The IAM policy document ARN."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy ARN with the -p parameter."
  usage
  return 1
fi

response=$(aws iam attach-role-policy \
  --role-name "$role_name" \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
  return 1
```

```
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous AttachRolePolicyà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::attachRolePolicy(const Aws::String &roleName,
                                   const Aws::String &policyArn,
                                   const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::ListAttachedRolePoliciesRequest list_request;
    list_request.SetRoleName(roleName);

    bool done = false;
    while (!done) {
        auto list_outcome = iam.ListAttachedRolePolicies(list_request);
        if (!list_outcome.IsSuccess()) {
            std::cerr << "Failed to list attached policies of role " <<
                      roleName << ": " << list_outcome.GetError().GetMessage() <<
                      std::endl;
            return false;
        }

        const auto &policies = list_outcome.GetResult().GetAttachedPolicies();
```

```
        if (std::any_of(policies.cbegin(), policies.cend(),
                        [=](const Aws::IAM::Model::AttachedPolicy &policy) {
                                return policy.GetPolicyArn() == policyArn;
                        })) {
            std::cout << "Policy " << policyArn <<
                      " is already attached to role " << roleName << std::endl;
            return true;
        }

        done = !list_outcome.GetResult().GetIsTruncated();
        list_request.SetMarker(list_outcome.GetResult().GetMarker());
    }

    Aws::IAM::Model::AttachRolePolicyRequest request;
    request.SetRoleName(roleName);
    request.SetPolicyArn(policyArn);

    Aws::IAM::Model::AttachRolePolicyOutcome outcome =
 iam.AttachRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to attach policy " << policyArn << " to role " <<
                  roleName << ": " << outcome.GetError().GetMessage() <<
 std::endl;
    }
    else {
        std::cout << "Successfully attached policy " << policyArn << " to role "
 <<
                  roleName << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, reportez-vous AttachRolePolicyà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour attacher une politique gérée à un rôle IAM

La `attach-role-policy` commande suivante associe la politique AWS gérée nommée `ReadOnlyAccess` au rôle IAM nomméReadOnlyRole.

```
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --role-name ReadOnlyRole
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Politiques gérées et politiques en ligne dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous AttachRolePolicyà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
```

```
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(ctx context.Context, policyArn
 string, roleName string) error {
 _, err := wrapper.IamClient.AttachRolePolicy(ctx, &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}
```

- Pour plus de détails sur l'API, reportez-vous AttachRolePolicyà la section Référence des AWS SDK pour Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.AttachRolePolicyRequest;
import software.amazon.awssdk.services.iam.model.AttachedPolicy;
import software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesRequest;
import
 software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesResponse;
```

```
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AttachRolePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleName = args[0];
        String policyArn = args[1];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        attachIAMRolePolicy(iam, roleName, policyArn);
        iam.close();
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
 policyArn) {
```

```
        try {
            ListAttachedRolePoliciesRequest request =
 ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();

            ListAttachedRolePoliciesResponse response =
 iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();

            // Ensure that the policy is not attached to this role
            String polArn = "";
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
 this role.");
                    return;
                }
            }

            AttachRolePolicyRequest attachRequest =
 AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);

            System.out.println("Successfully attached policy " + policyArn +
                    " to role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Pour plus de détails sur l'API, reportez-vous AttachRolePolicyà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Attachez la politique.

```
import { AttachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 * @param {string} roleName
 */
export const attachRolePolicy = (policyArn, roleName) => {
  const command = new AttachRolePolicyCommand({
    PolicyArn: policyArn,
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous AttachRolePolicyà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        console.log(
          "AmazonDynamoDBFullAccess is already attached to this role."
        );
        process.exit();
      }
    });
    var params = {
      PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
      RoleName: process.argv[2],
    };
    iam.attachRolePolicy(params, function (err, data) {
      if (err) {
        console.log("Unable to attach policy to role", err);
      } else {
        console.log("Role attached successfully");
      }
    });
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.

- Pour plus de détails sur l'API, reportez-vous AttachRolePolicyà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun attachIAMRolePolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
```

```
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String,
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}
```

- Pour plus de détails sur l'API, reportez-vous AttachRolePolicyà la section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
```

```
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

    public function attachRolePolicy($roleName, $policyArn)
    {
        return $this->customWaiter(function () use ($roleName, $policyArn) {
            $this->iamClient->attachRolePolicy([
                'PolicyArn' => $policyArn,
                'RoleName' => $roleName,
            ]);
        });
    }
```

- Pour plus de détails sur l'API, reportez-vous AttachRolePolicyà la section Référence des AWS SDK pour PHP API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple associe la politique AWS gérée nommée **SecurityAudit** au rôle **CoSecurityAuditors** IAM. Les utilisateurs qui assument ce rôle sont immédiatement affectés par les autorisations définies dans la version la plus récente de cette politique.

```
Register-IAMRolePolicy -RoleName CoSecurityAuditors -PolicyArn
 arn:aws:iam::aws:policy/SecurityAudit
```

- Pour plus de détails sur l'API, reportez-vous AttachRolePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple associe la politique AWS gérée nommée **SecurityAudit** au rôle **CoSecurityAuditors** IAM. Les utilisateurs qui assument ce rôle sont immédiatement affectés par les autorisations définies dans la version la plus récente de cette politique.

```
Register-IAMRolePolicy -RoleName CoSecurityAuditors -PolicyArn
 arn:aws:iam::aws:policy/SecurityAudit
```

- Pour plus de détails sur l'API, reportez-vous AttachRolePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Attacher une politique à un rôle à l'aide de l'objet de politique Boto3.

```
def attach_to_role(role_name, policy_arn):
```

```
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Policy(policy_arn).attach_role(RoleName=role_name)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise
```

Attacher une politique à un rôle à l'aide de l'objet de rôle Boto3.

```
def attach_policy(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise
```

- Pour plus de détails sur l'API, consultez AttachRolePolicyle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, attache et détache les politiques de rôle.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
```

```
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
```

```
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, reportez-vous AttachRolePolicyà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn attach_role_policy(
    client: &iamClient,
    role: &Role,
    policy: &Policy,
) -> Result<AttachRolePolicyOutput, SdkError<AttachRolePolicyError>> {
    client
        .attach_role_policy()
        .role_name(role.role_name())
        .policy_arn(policy.arn().unwrap_or_default())
        .send()
        .await
```

```
}
```

- Pour plus de détails sur l'API, voir AttachRolePolicyla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    lo_iam->attachrolepolicy(
      iv_rolename = iv_role_name
      iv_policyarn = iv_policy_arn ).
    MESSAGE 'Policy attached to role successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Role or policy does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Policy attachment limit exceeded.' TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous AttachRolePolicyà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func attachRolePolicy(role: String, policyArn: String) async throws {
        let input = AttachRolePolicyInput(
            policyArn: policyArn,
            roleName: role
        )
        do {
            _ = try await client.attachRolePolicy(input: input)
        } catch {
            print("ERROR: Attaching a role policy:", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous AttachRolePolicyà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`AttachUserPolicy`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `AttachUserPolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Créer des utilisateurs en lecture seule et en lecture-écriture

CLI

AWS CLI

Pour attacher une politique gérée à un utilisateur IAM

La `attach-user-policy` commande suivante associe la politique AWS gérée nommée `AdministratorAccess` à l'utilisateur IAM nomméAlice.

```
aws iam attach-user-policy \
    --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \
    --user-name Alice
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Politiques gérées et politiques en ligne dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous AttachUserPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple attache la politique AWS gérée nommée **AmazonCognitoPowerUser** à l'utilisateur **Bob** IAM. L'utilisateur est immédiatement affecté par les autorisations définies dans la version la plus récente de cette politique.

```
Register-IAMUserPolicy -UserName Bob -PolicyArn arn:aws:iam::aws:policy/
AmazonCognitoPowerUser
```

- Pour plus de détails sur l'API, reportez-vous AttachUserPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple attache la politique AWS gérée nommée **AmazonCognitoPowerUser** à l'utilisateur **Bob** IAM. L'utilisateur est immédiatement affecté par les autorisations définies dans la version la plus récente de cette politique.

```
Register-IAMUserPolicy -UserName Bob -PolicyArn arn:aws:iam::aws:policy/
AmazonCognitoPowerUser
```

- Pour plus de détails sur l'API, reportez-vous AttachUserPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
 user_name)
        raise
```

- Pour plus de détails sur l'API, consultez AttachUserPolicyle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
  # Attaches a policy to a user
  #
  # @param user_name [String] The name of the user
  # @param policy_arn [String] The Amazon Resource Name (ARN) of the policy
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_user(user_name, policy_arn)
    @iam_client.attach_user_policy(
      user_name: user_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to user: #{e.message}")
    false
  end
```

- Pour plus de détails sur l'API, reportez-vous AttachUserPolicyà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn attach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .attach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;
```

```
    Ok(())
}
```

- Pour plus de détails sur l'API, voir AttachUserPolicyla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    lo_iam->attachuserpolicy(
      iv_username = iv_user_name
      iv_policyarn = iv_policy_arn ).
    MESSAGE 'Policy attached to user successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'User or policy does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Policy attachment limit exceeded.' TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous AttachUserPolicyà la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`ChangePassword`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `ChangePassword`.

CLI

AWS CLI

Pour changer le mot de passe de votre utilisateur IAM

Pour modifier le mot de passe de votre utilisateur IAM, nous vous recommandons d'utiliser le paramètre `--cli-input-json` pour transmettre un fichier JSON contenant vos anciens et nouveaux mots de passe. Grâce à cette méthode, vous pouvez utiliser des mots de passe forts contenant des caractères non alphanumériques. Il peut être difficile d'utiliser des mots de passe contenant des caractères non alphanumériques lorsque vous les transmettez en tant que paramètres de ligne de commande. Pour utiliser le paramètre `--cli-input-json`, commencez par utiliser la commande `change-password` avec le paramètre `--generate-cli-skeleton`, comme dans l'exemple suivant.

```
aws iam change-password \
    --generate-cli-skeleton > change-password.json
```

La commande précédente crée un fichier JSON appelé change-password.json que vous pouvez utiliser pour renseigner vos anciens et nouveaux mots de passe. Par exemple, le fichier peut ressembler à ce qui suit.

```
{
    "OldPassword": "3s0K_;xh4~8XXI",
    "NewPassword": "]35d/{pB9Fo9wJ"
}
```

Ensuite, pour modifier votre mot de passe, réutilisez la commande `change-password`, en passant cette fois le paramètre `--cli-input-json` pour spécifier votre fichier JSON. La commande `change-password` suivante utilise le paramètre `--cli-input-json` avec un fichier JSON appelé change-password.json.

```
aws iam change-password \
    --cli-input-json file://change-password.json
```

Cette commande ne produit aucune sortie.

Cette commande ne peut être appelée que par les utilisateurs IAM. Si cette commande est appelée à l'aide des informations d'identification du AWS compte (root), elle renvoie une `InvalidUserType` erreur.

Pour plus d'informations, consultez Comment un utilisateur IAM modifie son propre mot de passe dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ChangePasswordà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cette commande modifie le mot de passe de l'utilisateur qui exécute la commande. Cette commande ne peut être appelée que par les utilisateurs IAM. Si cette commande est appelée lorsque vous êtes connecté avec les informations d'identification du AWS compte (root), elle renvoie une erreur. **`InvalidUserType`**

```
Edit-IAMPassword -OldPassword "MyOldP@ssw0rd" -NewPassword "MyNewP@ssw0rd"
```

- Pour plus de détails sur l'API, reportez-vous ChangePasswordà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cette commande modifie le mot de passe de l'utilisateur qui exécute la commande. Cette commande ne peut être appelée que par les utilisateurs IAM. Si cette commande est appelée lorsque vous êtes connecté avec les informations d'identification du AWS compte (root), elle renvoie une erreur. **`InvalidUserType`**

```
Edit-IAMPassword -OldPassword "MyOldP@ssw0rd" -NewPassword "MyNewP@ssw0rd"
```

- Pour plus de détails sur l'API, reportez-vous ChangePasswordà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`CreateAccessKey`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `CreateAccessKey`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Principes de base
- Créer des utilisateurs en lecture seule et en lecture-écriture
- Gérer des clés d'accès

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }
```

- Pour plus de détails sur l'API, reportez-vous CreateAccessKeyà la section Référence des AWS SDK pour .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
```

```
  echo "  -u user_name   The name of the IAM user."
  echo "  [-f file_name]   Optional file name for the access key output."
  echo ""
}

# Retrieve the calling parameters.
while getopts "u:f:h" option; do
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    f) file_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

response=$(aws iam create-access-key \
  --user-name "$user_name" \
  --output text)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-access-key operation failed.$response"
  return 1
fi

if [[ -n "$file_name" ]]; then
  echo "$response" >"$file_name"
fi
```

```
  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous CreateAccessKeyà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
Aws::String AwsDoc::IAM::createAccessKey(const Aws::String &userName,
                                         const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreateAccessKeyRequest request;
    request.SetUserName(userName);

    Aws::String result;
    Aws::IAM::Model::CreateAccessKeyOutcome outcome =
 iam.CreateAccessKey(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating access key for IAM user " << userName
                  << ":" << outcome.GetError().GetMessage() << std::endl;
    }
```

```
    else {
        const auto &accessKey = outcome.GetResult().GetAccessKey();
        std::cout << "Successfully created access key for IAM user " <<
                  userName << std::endl << "  aws_access_key_id = " <<
                  accessKey.GetAccessKeyId() << std::endl <<
                  " aws_secret_access_key = " << accessKey.GetSecretAccessKey()
 <<
                  std::endl;
        result = accessKey.GetAccessKeyId();
    }

    return result;
}
```

- Pour plus de détails sur l'API, reportez-vous CreateAccessKeyà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour créer une clé d'accès pour un utilisateur IAM

La commande `create-access-key` suivante créer une clé d'accès (un ID de clé d'accès et une clé d'accès secrète) pour l'utilisateur IAM nommé Bob.

```
aws iam create-access-key \
    --user-name Bob
```

Sortie :

```
{
    "AccessKey": {
        "UserName": "Bob",
        "Status": "Active",
        "CreateDate": "2015-03-09T18:39:23.411Z",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
}
```

Conservez votre clé d'accès secrète dans un emplacement sécurisé. Si elle est perdue, elle ne peut pas être récupérée et vous devez créer une nouvelle clé.

Pour plus d'informations, consultez Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous CreateAccessKeyà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
```

```
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(ctx context.Context, userName
 string) (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(ctx, &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}
```

- Pour plus de détails sur l'API, reportez-vous CreateAccessKeyà la section Référence des AWS SDK pour Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.CreateAccessKeyRequest;
import software.amazon.awssdk.services.iam.model.CreateAccessKeyResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
```

```
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAccessKey {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                   <user>\s

                Where:
                   user - An AWS IAM user that you can obtain from the AWS
 Management Console.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String user = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String keyId = createIAMAccessKey(iam, user);
        System.out.println("The Key Id is " + keyId);
        iam.close();
    }

    public static String createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey().accessKeyId();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
```

```
        }
        return "";
    }
}
```

- Pour plus de détails sur l'API, reportez-vous CreateAccessKeyà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez la clé d'accès.

```
import { CreateAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 */
export const createAccessKey = (userName) => {
  const command = new CreateAccessKeyCommand({ UserName: userName });
  return client.send(command);
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous CreateAccessKeyà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccessKey({ UserName: "IAM_USER_NAME" }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.AccessKey);
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous CreateAccessKeyà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun createIAMAccessKey(user: String?): String {
    val request =
        CreateAccessKeyRequest {
            userName = user
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createAccessKey(request)
        return response.accessKey?.accessKeyId.toString()
    }
}
```

- Pour plus de détails sur l'API, reportez-vous CreateAccessKeyà la section AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple crée une paire de clés d'accès et de clés d'accès secrètes et les attribue à l'utilisateur **David**. Prenez soin d'enregistrer les valeurs **AccessKeyId** et **SecretAccessKey** dans un fichier, car c'est la seule fois où vous pouvez obtenir les **SecretAccessKey**. Vous ne pourrez pas la récupérer ultérieurement. Si vous perdez votre clé d'accès secrète, vous devez créer une paire de clés.

```
New-IAMAccessKey -UserName David
```

Sortie :

```
AccessKeyId     : AKIAIOSFODNN7EXAMPLE
CreateDate      : 4/13/2015 1:00:42 PM
SecretAccessKey : wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Status          : Active
UserName        : David
```

- Pour plus de détails sur l'API, reportez-vous CreateAccessKeyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple crée une paire de clés d'accès et de clés d'accès secrètes et les attribue à l'utilisateur **David**. Prenez soin d'enregistrer les valeurs **AccessKeyId** et **SecretAccessKey** dans un fichier, car c'est la seule fois où vous pouvez obtenir les **SecretAccessKey**. Vous ne pourrez pas la récupérer ultérieurement. Si vous perdez votre clé d'accès secrète, vous devez créer une paire de clés.

```
New-IAMAccessKey -UserName David
```

Sortie :

```
AccessKeyId     : AKIAIOSFODNN7EXAMPLE
CreateDate      : 4/13/2015 1:00:42 PM
SecretAccessKey : wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Status          : Active
UserName        : David
```

- Pour plus de détails sur l'API, reportez-vous CreateAccessKeyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
```

```
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair
```

- Pour plus de détails sur l'API, consultez CreateAccessKeyle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, désactive et supprime les clés d'accès.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'AccessKeyManager'
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
```

```
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded
    @logger.error('Error creating access key: limit exceeded. Cannot create
more.')
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: 'Inactive'
    )
```

```
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, reportez-vous CreateAccessKeyà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn create_access_key(client: &iamClient, user_name: &str) ->
 Result<AccessKey, iamError> {
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;
```

```
    let response: Result<CreateAccessKeyOutput, SdkError<CreateAccessKeyError>> =
 loop {
        match client.create_access_key().user_name(user_name).send().await {
            Ok(inner_response) => {
                break Ok(inner_response);
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    Ok(response.unwrap().access_key.unwrap())
}
```

- Pour plus de détails sur l'API, voir CreateAccessKeyla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->createaccesskey(
          iv_username = iv_user_name ).
        MESSAGE 'Access key created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
```

```
        MESSAGE 'Maximum number of access keys reached.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous CreateAccessKeyà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func createAccessKey(userName: String) async throws ->
 IAMClientTypes.AccessKey {
        let input = CreateAccessKeyInput(
            userName: userName
        )
        do {
            let output = try await iamClient.createAccessKey(input: input)
            guard let accessKey = output.accessKey else {
                throw ServiceHandlerError.keyError
            }
            return accessKey
        } catch {
            print("ERROR: createAccessKey:", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateAccessKeyà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`CreateAccountAlias`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `CreateAccountAlias`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer votre compte

C++

SDK pour C++

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::createAccountAlias(const Aws::String &aliasName,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::CreateAccountAliasRequest request;
    request.SetAccountAlias(aliasName);

    Aws::IAM::Model::CreateAccountAliasOutcome outcome = iam.CreateAccountAlias(
            request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating account alias " << aliasName << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
```

```
    else {
        std::cout << "Successfully created account alias " << aliasName <<
                  std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, voir CreateAccountAliasla section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour créer un alias de compte

La `create-account-alias` commande suivante crée l'alias `examplecorp` de votre AWS compte.

```
aws iam create-account-alias \
    --account-alias examplecorp
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez la section Votre identifiant de AWS compte et son alias dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir CreateAccountAliasla section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.CreateAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAccountAlias {
    public static void main(String[] args) {
        final String usage = """
                Usage:
                    <alias>\s

                Where:
                    alias - The account alias to create (for example,
 myawsaccount).\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        createIAMAccountAlias(iam, alias);
        iam.close();
        System.out.println("Done");
    }

    public static void createIAMAccountAlias(IamClient iam, String alias) {
        try {
```

```
            CreateAccountAliasRequest request =
 CreateAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();

            iam.createAccountAlias(request);
            System.out.println("Successfully created account alias: " + alias);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, voir CreateAccountAliasla section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez l'alias de compte.

```
import { CreateAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} alias - A unique name for the account alias.
 * @returns
 */
export const createAccountAlias = (alias) => {
```

```
  const command = new CreateAccountAliasCommand({
    AccountAlias: alias,
  });

  return client.send(command);
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, voir CreateAccountAliasla section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, voir CreateAccountAliasla section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun createIAMAccountAlias(alias: String) {
    val request =
        CreateAccountAliasRequest {
            accountAlias = alias
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.createAccountAlias(request)
        println("Successfully created account alias named $alias")
    }
}
```

- Pour plus de détails sur l'API, consultez CreateAccountAliasla section AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Dans cet exemple, l'alias de votre AWS compte est remplacé par**mycompanyaws**. <accountidnumber>L'adresse de la page de connexion de l'utilisateur devient https://mycompanyaws.signin.aws.amazon.com/console. l'URL d'origine en utilisant votre numéro d'identification de compte au lieu de l'alias (https ://.signin.aws.amazon). com/console) continue de fonctionner. Cependant, toutes les URL basées sur des alias définies précédemment cessent de fonctionner.

```
New-IAMAccountAlias -AccountAlias mycompanyaws
```

- Pour plus de détails sur l'API, reportez-vous CreateAccountAliasà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Dans cet exemple, l'alias de votre AWS compte est remplacé par**mycompanyaws**. <accountidnumber>L'adresse de la page de connexion de l'utilisateur devient https://mycompanyaws.signin.aws.amazon.com/console. l'URL d'origine en utilisant votre numéro d'identification de compte au lieu de l'alias (https ://.signin.aws.amazon). com/console) continue de fonctionner. Cependant, toutes les URL basées sur des alias définies précédemment cessent de fonctionner.

```
New-IAMAccountAlias -AccountAlias mycompanyaws
```

- Pour plus de détails sur l'API, reportez-vous CreateAccountAliasà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
```

```
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
        raise
```

- Pour plus de détails sur l'API, consultez CreateAccountAliasle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Listez, créez et supprimez les alias de compte.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info('Account aliases are:')
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info('No account aliases found.')
    end
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, voir CreateAccountAliasla section Référence des AWS SDK pour Ruby API.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        lo_iam->createaccountalias(
          iv_accountalias = iv_account_alias ).
        MESSAGE 'Account alias created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'Account alias already exists.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Account alias limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, consultez CreateAccountAliasla section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`CreateGroup`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `CreateGroup`.

CLI

AWS CLI

Pour créer un groupe IAM

La commande `create-group` suivante crée un groupe IAM nommé `Admins`.

```
aws iam create-group \
    --group-name Admins
```

Sortie :

```
{
    "Group": {
        "Path": "/",
        "CreateDate": "2015-03-09T20:30:24.940Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
```

```
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "GroupName": "Admins"
    }
}
```

Pour plus d'informations, consultez Création de groupes d'utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous CreateGroupà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { CreateGroupCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} groupName
 */
export const createGroup = async (groupName) => {
  const command = new CreateGroupCommand({ GroupName: groupName });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Pour plus de détails sur l'API, reportez-vous CreateGroupà la section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple crée un groupe IAM nommé **`Developers`**.

```
New-IAMGroup -GroupName Developers
```

Sortie :

```
Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 4/14/2015 11:21:31 AM
GroupId    : QNEJ5PM4NFSQCEXAMPLE1
GroupName  : Developers
Path       : /
```

- Pour plus de détails sur l'API, reportez-vous CreateGroupà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple crée un groupe IAM nommé **`Developers`**.

```
New-IAMGroup -GroupName Developers
```

Sortie :

```
Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 4/14/2015 11:21:31 AM
GroupId    : QNEJ5PM4NFSQCEXAMPLE1
GroupName  : Developers
Path       : /
```

- Pour plus de détails sur l'API, reportez-vous CreateGroupà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`CreateInstanceProfile`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `CreateInstanceProfile`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Créer et gérer un service résilient
- Création d'un service de conteneur pour les instances de machines virtuelles
- Débuter avec le streaming géré
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /// <summary>
   /// Create a policy, role, and profile that is associated with instances with
a specified name.
   /// An instance's associated profile defines a role that is assumed by the
   /// instance.The role has attached policies that specify the AWS permissions
granted to
   /// clients that run on the instance.
   /// </summary>
   /// <param name="policyName">Name to use for the policy.</param>
   /// <param name="roleName">Name to use for the role.</param>
   /// <param name="profileName">Name to use for the profile.</param>
   /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
   /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
   /// <returns>The Arn of the profile.</returns>
   public async Task<string> CreateInstanceProfileWithName(
```

```
        string policyName,
        string roleName,
        string profileName,
        string ssmOnlyPolicyFile,
        List<string>? awsManagedPolicies = null)
    {

        var assumeRoleDoc = "{" +
                                    "\"Version\": \"2012-10-17\"," +
                                    "\"Statement\": [{" +
                                         "\"Effect\": \"Allow\"," +
                                         "\"Principal\": {" +
                                         "\"Service\": [" +
                                             "\"ec2.amazonaws.com\"" +
                                         "]" +
                                         "}," +
                                    "\"Action\": \"sts:AssumeRole\"" +
                                    "}]" +
                                "}";

        var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

        var policyArn = "";

        try
        {
            var createPolicyResult = await _amazonIam.CreatePolicyAsync(
                new CreatePolicyRequest
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
```

```
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
```

```
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateInstanceProfileà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour créer un profil d'instance

La commande `create-instance-profile` suivante crée un profil d'instance nommé `Webserver`.

```
aws iam create-instance-profile \
    --instance-profile-name Webserver
```

Sortie :

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AIPAJMBYC7DLSPEXAMPLE",
        "Roles": [],
        "CreateDate": "2015-03-09T20:33:19.626Z",
        "InstanceProfileName": "Webserver",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:instance-profile/Webserver"
    }
}
```

Pour ajouter un rôle à un profil d'instance, utilisez la commande `add-role-to-instance-profile`.

Pour plus d'informations, consultez Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2 dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous CreateInstanceProfileà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
  const { InstanceProfile } = await iamClient.send(
```

```
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
  await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
```

- Pour plus de détails sur l'API, reportez-vous CreateInstanceProfileà la section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple crée un profil d'instance IAM nommé **ProfileForDevEC2Instance**. Vous devez exécuter la commande **Add-IAMRoleToInstanceProfile** séparément pour associer le profil d'instance à un rôle IAM existant qui fournit des autorisations à l'instance. Pour finir, attachez le profil d'instance à une instance EC2 lorsque vous la lancez. Pour ce faire, utilisez l'applet de commande **New-EC2Instance** avec le paramètre **InstanceProfile_Arn** ou **InstanceProfile_Name**.

```
New-IAMInstanceProfile -InstanceProfileName ProfileForDevEC2Instance
```

Sortie :

```
Arn                 : arn:aws:iam::123456789012:instance-profile/
ProfileForDevEC2Instance
CreateDate          : 4/14/2015 11:31:39 AM
InstanceProfileId   : DYMFXL556EY46EXAMPLE1
InstanceProfileName : ProfileForDevEC2Instance
Path                : /
Roles               : {}
```

- Pour plus de détails sur l'API, reportez-vous CreateInstanceProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple crée un profil d'instance IAM nommé **`ProfileForDevEC2Instance`**. Vous devez exécuter la commande **`Add-IAMRoleToInstanceProfile`** séparément pour associer le profil d'instance à un rôle IAM existant qui fournit des autorisations à l'instance. Pour finir, attachez le profil d'instance à une instance EC2 lorsque vous la lancez. Pour ce faire, utilisez l'applet de commande **`New-EC2Instance`** avec le paramètre **`InstanceProfile_Arn`** ou **`InstanceProfile_Name`**.

```
New-IAMInstanceProfile -InstanceProfileName ProfileForDevEC2Instance
```

Sortie :

```
Arn                 : arn:aws:iam::123456789012:instance-profile/
ProfileForDevEC2Instance
CreateDate          : 4/14/2015 11:31:39 AM
InstanceProfileId   : DYMFXL556EY46EXAMPLE1
InstanceProfileName : ProfileForDevEC2Instance
Path                : /
Roles               : {}
```

- Pour plus de détails sur l'API, reportez-vous CreateInstanceProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple crée une politique, un rôle et un profil d'instance et les lie tous ensemble.

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """
```

```
   def __init__(
       self,
       resource_prefix: str,
       inst_type: str,
       ami_param: str,
       autoscaling_client: boto3.client,
       ec2_client: boto3.client,
       ssm_client: boto3.client,
       iam_client: boto3.client,
   ):
       """
       Initializes the AutoScaler class with the necessary parameters.

       :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
       :param inst_type: The type of EC2 instance to create, such as t3.micro.
       :param ami_param: The Systems Manager parameter used to look up the AMI
that is created.
       :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
       :param ec2_client: A Boto3 EC2 client.
       :param ssm_client: A Boto3 Systems Manager client.
       :param iam_client: A Boto3 IAM client.
       """
       self.inst_type = inst_type
       self.ami_param = ami_param
       self.autoscaling_client = autoscaling_client
       self.ec2_client = ec2_client
       self.ssm_client = ssm_client
       self.iam_client = iam_client
       sts_client = boto3.client("sts")
       self.account_id = sts_client.get_caller_identity()["Account"]

       self.key_pair_name = f"{resource_prefix}-key-pair"
       self.launch_template_name = f"{resource_prefix}-template-"
       self.group_name = f"{resource_prefix}-group"

       # Happy path
       self.instance_policy_name = f"{resource_prefix}-pol"
       self.instance_role_name = f"{resource_prefix}-role"
       self.instance_profile_name = f"{resource_prefix}-prof"

       # Failure mode
       self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
```

```
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def create_instance_profile(
        self,
        policy_file: str,
        policy_name: str,
        role_name: str,
        profile_name: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> str:
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
```

```
        }
        policy_arn = self.create_policy(policy_file, policy_name)
        self.create_role(role_name, assume_role_doc)
        self.attach_policy(role_name, policy_arn, aws_managed_policies)

        try:
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            log.error(f"Full error:\n\t{err}")
        return profile_arn
```

- Pour plus de détails sur l'API, consultez CreateInstanceProfilele AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`CreateLoginProfile`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `CreateLoginProfile`.

CLI

AWS CLI

Pour générer le mot de passe d'un utilisateur IAM

Pour générer le mot de passe d'un utilisateur IAM, nous vous recommandons d'utiliser le paramètre `--cli-input-json` pour transmettre un fichier JSON contenant le mot de passe. Grâce à cette méthode, vous pouvez générer un mot de passe fort contenant des caractères non alphanumériques. Il peut être difficile de générer un mot de passe contenant des caractères non alphanumériques lorsque vous le transmettez en tant que paramètre de la ligne de commande.

Pour utiliser le paramètre `--cli-input-json`, commencez par utiliser la commande `create-login-profile` avec le paramètre `--generate-cli-skeleton`, comme dans l'exemple suivant.

```
aws iam create-login-profile \
    --generate-cli-skeleton > create-login-profile.json
```

La commande précédente crée un fichier JSON appelé create-login-profile.json que vous pouvez utiliser pour renseigner les informations d'une commande `create-login-profile` ultérieure. Par exemple :

```
{
    "UserName": "Bob",
    "Password": "&1-3a6u:RA0djs",
    "PasswordResetRequired": true
}
```

Ensuite, pour générer le mot de passe d'un utilisateur IAM, réutilisez la commande `create-login-profile`, en passant cette fois le paramètre `--cli-input-json` pour spécifier votre fichier JSON. La commande `create-login-profile` suivante utilise le paramètre `--cli-input-json` avec un fichier JSON appelé create-login-profile.json.

```
aws iam create-login-profile \
```

```
    --cli-input-json file://create-login-profile.json
```

Sortie :

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2015-03-10T20:55:40.274Z",
        "PasswordResetRequired": true
    }
}
```

Si le nouveau mot de passe enfreint la politique de mot de passe du compte, la commande renvoie une erreur `PasswordPolicyViolation`.

Pour modifier le mot de passe d'un utilisateur qui en possède déjà un, utilisez `update-login-profile`. Pour définir une politique de mot de passe pour le compte, utilisez la commande `update-account-password-policy`.

Si la politique de mot de passe du compte le permet, les utilisateurs IAM peuvent modifier leurs propres mots de passe à l'aide de la commande `change-password`.

Pour plus d'informations, consultez Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous CreateLoginProfileà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple crée un mot de passe (temporaire) pour l'utilisateur IAM nommé Bob et définit l'indicateur qui oblige l'utilisateur à modifier le mot de passe lors de la prochaine connexion de **Bob**.

```
New-IAMLoginProfile -UserName Bob -Password P@ssw0rd -PasswordResetRequired $true
```

Sortie :

```
CreateDate                    PasswordResetRequired                UserName
```

```
----------                  ---------------------           --------
4/14/2015 12:26:30 PM       True                            Bob
```

- Pour plus de détails sur l'API, reportez-vous CreateLoginProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple crée un mot de passe (temporaire) pour l'utilisateur IAM nommé Bob et définit l'indicateur qui oblige l'utilisateur à modifier le mot de passe lors de la prochaine connexion de **Bob**.

```
New-IAMLoginProfile -UserName Bob -Password P@ssw0rd -PasswordResetRequired $true
```

Sortie :

```
CreateDate                  PasswordResetRequired           UserName
----------                  ---------------------           --------
4/14/2015 12:26:30 PM       True                            Bob
```

- Pour plus de détails sur l'API, reportez-vous CreateLoginProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`CreateOpenIdConnectProvider`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `CreateOpenIdConnectProvider`.

CLI

AWS CLI

Pour créer un fournisseur OpenID Connect (OIDC)

Pour créer un fournisseur OpenID Connect (OIDC), nous vous recommandons d'utiliser le paramètre `--cli-input-json` pour transmettre un fichier JSON contenant les paramètres requis. Lorsque vous créez un fournisseur OIDC, vous devez transmettre l'URL du fournisseur, qui doit commencer par `https://`. Il peut être difficile de transmettre l'URL en tant que

paramètre de ligne de commande, car les deux points (:) et les barres obliques (/) ont une signification particulière dans certains environnements de ligne de commande. L'utilisation du paramètre `--cli-input-json` permet de contourner cette limitation.

Pour utiliser le paramètre `--cli-input-json`, commencez par utiliser la commande `create-open-id-connect-provider` avec le paramètre `--generate-cli-skeleton`, comme dans l'exemple suivant.

```
aws iam create-open-id-connect-provider \
    --generate-cli-skeleton > create-open-id-connect-provider.json
```

La commande précédente crée un fichier JSON appelé create-open-id-connect-provider.json que vous pouvez utiliser pour renseigner les informations d'une commande `create-open-id-connect-provider` ultérieure. Par exemple :

```
{
    "Url": "https://server.example.com",
    "ClientIDList": [
        "example-application-ID"
    ],
    "ThumbprintList": [
        "c3768084dfb3d2b68b7897bf5f565da8eEXAMPLE"
    ]
}
```

Si les points de terminaison JWKS et de découverte de votre fournisseur OIDC utilisent des certificats différents, incluez les empreintes digitales des deux points de terminaison dans le. `ThumbprintList`

Ensuite, pour créer le fournisseur OpenID Connect (OIDC), réutilisez la commande `create-open-id-connect-provider`, en passant cette fois le paramètre `--cli-input-json` pour spécifier votre fichier JSON. La commande `create-open-id-connect-provider` suivante utilise le paramètre `--cli-input-json` avec un fichier JSON appelé create-open-id-connect-provider.json.

```
aws iam create-open-id-connect-provider \
    --cli-input-json file://create-open-id-connect-provider.json
```

Sortie :

```
{
    "OpenIDConnectProviderArn": "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
}
```

Pour obtenir les informations sur les fournisseurs OIDC, consultez Création de fournisseurs d'identité OpenID Connect (OIDC) dans le Guide de l'utilisateur AWS IAM.

Pour plus d'informations sur l'obtention d'empreintes numériques pour un fournisseur OIDC, consultez Obtention de l'empreinte numérique pour un fournisseur d'identité OpenID Connect dans le Guide de l'utilisateur AWS IAM

- Pour plus de détails sur l'API, reportez-vous CreateOpenIdConnectProviderà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple crée un fournisseur IAM OIDC associé au service de fournisseur compatible OIDC figurant sur l'URL **`https://example.oidcprovider.com`** et l'ID client **`my-testapp-1`**. Le fournisseur OIDC fournit l'empreinte numérique. Pour authentifier l'empreinte numérique, suivez les étapes décrites dans http://docs.aws.amazon.com/IAM/latest/UserGuide/identity-providers-oidc-obtain-thumbprint.html.

```
New-IAMOpenIDConnectProvider -Url https://example.oidcprovider.com -ClientIDList
 my-testapp-1 -ThumbprintList 990F419EXAMPLEECF12DDEDA5EXAMPLE52F20D9E
```

Sortie :

```
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Pour plus de détails sur l'API, reportez-vous CreateOpenIdConnectProviderà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple crée un fournisseur IAM OIDC associé au service de fournisseur compatible OIDC figurant sur l'URL **`https://example.oidcprovider.com`** et l'ID client

my-testapp-1. Le fournisseur OIDC fournit l'empreinte numérique. Pour authentifier l'empreinte numérique, suivez les étapes décrites dans http://docs.aws.amazon.com/IAM/latest/UserGuide/identity-providers-oidc-obtain-thumbprint.html.

```
New-IAMOpenIDConnectProvider -Url https://example.oidcprovider.com -ClientIDList
 my-testapp-1 -ThumbprintList 990F419EXAMPLEECF12DDEDA5EXAMPLE52F20D9E
```

Sortie :

```
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Pour plus de détails sur l'API, reportez-vous CreateOpenIdConnectProviderà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **CreatePolicy**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `CreatePolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Principes de base
- Créer des utilisateurs en lecture seule et en lecture-écriture
- Création d'un espace de travail de surveillance géré
- Débuter avec le streaming géré
- Commencer à orchestrer les flux de travail
- Gérer les politiques
- Configurer le contrôle Attribute-Based d'accès
- Configuration de la gestion des systèmes
- Utiliser l'API IAM Policy Builder

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK pour .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
```

```
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) policy_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_name" ]]; then
  errecho "ERROR: You must provide a policy name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-policy \
  --policy-name "$policy_name" \
  --policy-document "$policy_document" \
  --output text \
  --query Policy.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-policy operation failed.\n$response"
```

```
    return 1
  fi

  echo "$response"
}
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
Aws::String AwsDoc::IAM::createPolicy(const Aws::String &policyName,
                                      const Aws::String &rsrcArn,
                                      const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreatePolicyRequest request;
    request.SetPolicyName(policyName);
    request.SetPolicyDocument(BuildSamplePolicyDocument(rsrcArn));

    Aws::IAM::Model::CreatePolicyOutcome outcome = iam.CreatePolicy(request);
    Aws::String result;
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating policy " << policyName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        result = outcome.GetResult().GetPolicy().GetArn();
        std::cout << "Successfully created policy " << policyName <<
                  std::endl;
    }
```

```
    return result;
}

Aws::String AwsDoc::IAM::BuildSamplePolicyDocument(const Aws::String &rsrc_arn) {
    std::stringstream stringStream;
    stringStream << "{"
                 << "  \"Version\": \"2012-10-17\","
                 << "  \"Statement\": ["
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": \"logs:CreateLogGroup\","
                 << "        \"Resource\": \""
                 << rsrc_arn
                 << "\""
                 << "    },"
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": ["
                 << "            \"dynamodb:DeleteItem\","
                 << "            \"dynamodb:GetItem\","
                 << "            \"dynamodb:PutItem\","
                 << "            \"dynamodb:Scan\","
                 << "            \"dynamodb:UpdateItem\""
                 << "       ],"
                 << "       \"Resource\": \""
                 << rsrc_arn
                 << "\""
                 << "    }"
                 << "   ]"
                 << "}";

    return stringStream.str();
}
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Exemple 1 : pour créer une politique gérée par le client

La commande suivante crée une politique gérée par le client nommée `my-policy`. Le fichier `policy.json` est un document JSON dans le dossier actuel qui accorde un accès en lecture seule au dossier `shared` dans un compartiment Amazon S3 nommé `amzn-s3-demo-bucket`.

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json
```

Contenu de policy.json

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/shared/*"
            ]
        }
    ]
}
```

Sortie :

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "CreateDate": "2015-06-01T19:31:18.620Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "ZXR6A36LTYANPAI7NJ5UV",
```

```
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::0123456789012:policy/my-policy",
        "UpdateDate": "2015-06-01T19:31:18.620Z"
    }
}
```

Pour plus d'informations sur l'utilisation de fichiers comme entrée pour les paramètres de chaîne, voir Spécifier les valeurs des paramètres pour la AWS CLI dans le guide de l'utilisateur de la AWS CLI.

Exemple 2 : pour créer une politique gérée par le client avec une description

La commande suivante crée une politique gérée par le client nommée `my-policy` avec une description immuable.

Le fichier `policy.json` est un document JSON dans le dossier actuel qui accorde un accès à toutes les actions Pull, List et Get dans un compartiment Amazon S3 nommé `amzn-s3-demo-bucket`.

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --description "This policy grants access to all Put, Get, and List actions
 for amzn-s3-demo-bucket"
```

Contenu de policy.json

```
{
   "Version":"2012-10-17",
   "Statement": [
       {
           "Effect": "Allow",
           "Action": [
                "s3:ListBucket*",
                "s3:PutBucket*",
                "s3:GetBucket*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        }
```

```
    ]
}
```

Sortie :

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T22:38:47+00:00",
        "UpdateDate": "2023-05-24T22:38:47+00:00"
    }
}
```

Pour plus d'informations sur Idenity-based les politiques, consultez les Identity-based politiques et les politiques basées sur les ressources dans le guide de l'utilisateur AWS IAM.

Exemple 3 : pour créer une politique gérée par le client avec des balises

La commande suivante crée une politique gérée par le client nommée `my-policy` avec des balises. Cet exemple utilise le `--tags` paramètre avec les JSON-formatted balises suivantes :`'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Le paramètre `--tags` peut également être utilisé avec des balises au format abrégé : `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`.

Le fichier `policy.json` est un document JSON dans le dossier actuel qui accorde un accès à toutes les actions Pull, List et Get dans un compartiment Amazon S3 nommé `amzn-s3-demo-bucket`.

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Contenu de policy.json

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket*",
                "s3:PutBucket*",
                "s3:GetBucket*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        }
    ]
}
```

Sortie :

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::12345678012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T23:16:39+00:00",
        "UpdateDate": "2023-05-24T23:16:39+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
                "Key": "Location",
                "Value": "Seattle"
            {
        ]
    }
```

```
}
```

Pour plus d'informations sur les politiques de balisage, consultez Balisage des politiques gérées par le client dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
```

```
 Version   string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect    string
 Action    []string
 Principal map[string]string `json:",omitempty"`
 Resource  *string           `json:",omitempty"`
}

// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(ctx context.Context, policyName string,
 actions []string,
 resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(ctx, &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
```

```
 return policy, err
}
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK pour Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreatePolicyRequest;
import software.amazon.awssdk.services.iam.model.CreatePolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyRequest;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.waiters.IamWaiter;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreatePolicy {

    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
```

```
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"dynamodb:DeleteItem\"," +
            "            \"dynamodb:GetItem\"," +
            "            \"dynamodb:PutItem\"," +
            "            \"dynamodb:Scan\"," +
            "            \"dynamodb:UpdateItem\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static void main(String[] args) {

        final String usage = """
                Usage:
                    CreatePolicy <policyName>\s

                Where:
                    policyName - A unique policy name.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String policyName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMPolicy(iam, policyName);
        System.out.println("Successfully created a policy with this ARN value: "
+ result);
        iam.close();
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
```

```
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument)
                    .build();

            CreatePolicyResponse response = iam.createPolicy(request);

            // Wait until the policy is created.
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
 iamWaiter.waitUntilPolicyExists(polRequest);

 waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez la politique.

```
import { CreatePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyName
 */
export const createPolicy = (policyName) => {
  const command = new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: "*",
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  });

  return client.send(command);
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
```

```
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var myManagedPolicy = {
  Version: "2012-10-17",
  Statement: [
    {
      Effect: "Allow",
      Action: "logs:CreateLogGroup",
      Resource: "RESOURCE_ARN",
    },
    {
      Effect: "Allow",
      Action: [
        "dynamodb:DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:PutItem",
        "dynamodb:Scan",
        "dynamodb:UpdateItem",
      ],
      Resource: "RESOURCE_ARN",
    },
  ],
};

var params = {
  PolicyDocument: JSON.stringify(myManagedPolicy),
  PolicyName: "myDynamoDBPolicy",
};

iam.createPolicy(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun createIAMPolicy(policyNameVal: String?): String {
    val policyDocumentVal = """
    {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "dynamodb:DeleteItem",
                    "dynamodb:GetItem",
                    "dynamodb:PutItem",
                    "dynamodb:Scan",
                    "dynamodb:UpdateItem"
                ],
                "Resource": "*"
            }
        ]
    }
    """.trimIndent()

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
```

```
        return response.policy?.arn.toString()
    }
}
```

- Pour plus de détails sur l'API, consultez CreatePolicyla section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

    /**
     * @param string $policyName
     * @param string $policyDocument
     * @return array
     */
    public function createPolicy(string $policyName, string $policyDocument)
    {
        $result = $this->customWaiter(function () use ($policyName,
 $policyDocument) {
```

```
            return $this->iamClient->createPolicy([
                'PolicyName' => $policyName,
                'PolicyDocument' => $policyDocument,
            ]);
        });
        return $result['Policy'];
    }
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK pour PHP API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple crée une nouvelle stratégie IAM dans le AWS compte courant nommée Le fichier **`MySamplePolicy.json`** fournit **`MySamplePolicy`** le contenu de la stratégie. Notez que vous devez utiliser le paramètre booléen **`-Raw`** pour traiter correctement le fichier de politique JSON.

```
New-IAMPolicy -PolicyName MySamplePolicy -PolicyDocument (Get-Content -Raw
 MySamplePolicy.json)
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 4/14/2015 2:45:59 PM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : LD4KP6HVFE7WGEXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 4/14/2015 2:45:59 PM
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple crée une nouvelle stratégie IAM dans le AWS compte courant nommée Le fichier **MySamplePolicy.json** fournit **MySamplePolicy** le contenu de la stratégie. Notez que vous devez utiliser le paramètre booléen **-Raw** pour traiter correctement le fichier de politique JSON.

```
New-IAMPolicy -PolicyName MySamplePolicy -PolicyDocument (Get-Content -Raw
 MySamplePolicy.json)
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 4/14/2015 2:45:59 PM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : LD4KP6HVFE7WGEXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 4/14/2015 2:45:59 PM
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.
```

```
    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::amzn-s3-demo-bucket/*' to allow actions on
all objects
                         in the bucket named 'amzn-s3-demo-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy
```

- Pour plus de détails sur l'API, consultez CreatePolicyle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, attache et détache les politiques de rôle.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
```

```
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
```

```
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn create_policy(
    client: &iamClient,
    policy_name: &str,
    policy_document: &str,
) -> Result<Policy, iamError> {
    let policy = client
        .create_policy()
        .policy_name(policy_name)
        .policy_document(policy_document)
        .send()
        .await?;
```

```
    Ok(policy.policy.unwrap())
}
```

- Pour plus de détails sur l'API, voir CreatePolicyla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->createpolicy(
          iv_policyname = iv_policy_name
          iv_policydocument = iv_policy_document
          iv_description = iv_description ).
        MESSAGE 'Policy created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'Policy already exists.' TYPE 'E'.
      CATCH /aws1/cx_iammalformedplydocex.
        MESSAGE 'Policy document is malformed.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Policy limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func createPolicy(name: String, policyDocument: String) async throws -
> IAMClientTypes.Policy {
        let input = CreatePolicyInput(
            policyDocument: policyDocument,
            policyName: name
        )
        do {
            let output = try await iamClient.createPolicy(input: input)
            guard let policy = output.policy else {
                throw ServiceHandlerError.noSuchPolicy
            }
            return policy
        } catch {
            print("ERROR: createPolicy:", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`CreatePolicyVersion`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `CreatePolicyVersion`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer les politiques

CLI

AWS CLI

Pour créer une version de la politique gérée

Cet exemple crée une nouvelle version `v2` de la politique IAM dont l'ARN est `arn:aws:iam::123456789012:policy/MyPolicy` et en fait la version par défaut.

```
aws iam create-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --policy-document file://NewPolicyVersion.json \
    --set-as-default
```

Sortie :

```
{
    "PolicyVersion": {
        "CreateDate": "2015-06-16T18:56:03.721Z",
        "VersionId": "v2",
        "IsDefaultVersion": true
    }
}
```

Pour plus d'informations, consultez Gestion des versions des politiques IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous CreatePolicyVersionà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple crée une version « v2 » de la politique IAM dont l'ARN est **`arn:aws:iam::123456789012:policy/MyPolicy`** et en fait la version par défaut. Le fichier **`NewPolicyVersion.json`** fournit le contenu de la politique. Notez que vous devez utiliser le paramètre booléen **`-Raw`** pour traiter correctement le fichier de politique JSON.

```
New-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy -
PolicyDocument (Get-content -Raw NewPolicyVersion.json) -SetAsDefault $true
```

Sortie :

```
CreateDate                          Document                 IsDefaultVersion
          VersionId
----------                          --------                 ----------------
          ---------
4/15/2015 10:54:54 AM                                        True
          v2
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyVersionà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple crée une version « v2 » de la politique IAM dont l'ARN est **`arn:aws:iam::123456789012:policy/MyPolicy`** et en fait la version par défaut. Le fichier **`NewPolicyVersion.json`** fournit le contenu de la politique. Notez que vous devez utiliser le paramètre booléen **`-Raw`** pour traiter correctement le fichier de politique JSON.

```
New-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy -
PolicyDocument (Get-content -Raw NewPolicyVersion.json) -SetAsDefault $true
```

Sortie :

```
CreateDate                          Document                 IsDefaultVersion
          VersionId
----------                          --------                 ----------------
          ---------
```

```
4/15/2015 10:54:54 AM                                                  True
           v2
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyVersionà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
            "Created policy version %s for policy %s.",
```

```
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
        logger.exception("Couldn't create a policy version for %s.", policy_arn)
        raise
    else:
        return policy_version
```

- Pour plus de détails sur l'API, consultez CreatePolicyVersionle AWS manuel de référence de l'API SDK for Python (Boto3).

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    oo_result = lo_iam->createpolicyversion(
      iv_policyarn = iv_policy_arn
      iv_policydocument = iv_policy_document
      iv_setasdefault = iv_set_as_default ).
    MESSAGE 'Policy version created successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Policy does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iammalformedplydocex.
    MESSAGE 'Policy document is malformed.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Policy version limit exceeded.' TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyVersionà la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`CreateRole`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `CreateRole`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Principes de base
- Configuration de la connectivité des services de conteneurs
- Création d'une tâche de conteneur pour le type de lancement sans serveur
- Création d'une API REST avec intégration de proxy de fonctions
- Création d'un service de conteneur pour les instances de machines virtuelles
- Création d'un espace de travail de surveillance géré
- Création d'un tableau de bord de surveillance avec le nom de la fonction comme variable
- Création de votre première fonction sans serveur
- Premiers pas avec les entrepôts de données sans serveur
- Commencer à gérer la configuration
- Débuter avec la protection des appareils liés à l'Internet des objets
- Commencer à utiliser les magasins de fonctionnalités d'apprentissage automatique
- Commencer à utiliser les clusters Kubernetes gérés
- Débuter avec le streaming géré
- Commencer à utiliser des clusters d'entrepôts de données provisionnés
- Commencer à orchestrer les flux de travail
- Gérer les rôles
- Transférer les secrets codés en dur vers un stockage secret sécurisé

- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts
- Configuration de la gestion des systèmes
- Utilisation de variables de propriété dans les tableaux de bord de surveillance pour surveiller plusieurs fonctions sans serveur
- Travaillez avec Streams et Time-to-Live

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS SDK pour .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.
```

```
# bashsupport disable=BP5008
function usage() {
  echo "function iam_create_user_access_key"
  echo "Creates an AWS Identity and Access Management (IAM) role."
  echo "  -n role_name   The name of the IAM role."
  echo "  -p policy_json -- The assume role policy document."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-role \
  --role-name "$role_name" \
  --assume-role-policy-document "$policy_document" \
  --output text \
  --query Role.Arn)
```

```
  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::createIamRole(
        const Aws::String &roleName,
        const Aws::String &policy,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::CreateRoleRequest request;

    request.SetRoleName(roleName);
    request.SetAssumeRolePolicyDocument(policy);

    Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating role. " <<
```

```
                    outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const Aws::IAM::Model::Role iamRole = outcome.GetResult().GetRole();
        std::cout << "Created role " << iamRole.GetRoleName() << "\n";
        std::cout << "ID: " << iamRole.GetRoleId() << "\n";
        std::cout << "ARN: " << iamRole.GetArn() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Exemple 1 : pour créer un rôle IAM

La commande `create-role` suivante crée un rôle nommé `Test-Role` et lui attache une politique d'approbation.

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json
```

Sortie :

```
{
    "Role": {
        "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "CreateDate": "2013-06-07T20:43:32.821Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

La politique de confiance est définie sous la forme d'un document JSON dans le Test-Role-Trust-Policy.jsonfichier. (Le nom et l'extension du fichier n'ont aucune importance.) La politique d'approbation doit spécifier un principal.

Pour attacher une politique des autorisations à un rôle, utilisez la commande `put-role-policy`.

Pour plus d'informations, consultez Création de rôles IAM dans le Guide de l'utilisateur AWS IAM.

Exemple 2 : pour créer un rôle IAM avec une durée de session maximale spécifiée

La commande `create-role` suivante crée un rôle nommé `Test-Role` et définit une durée de session maximale de 7 200 secondes (2 heures).

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --max-session-duration 7200
```

Sortie :

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:role/Test-Role",
        "CreateDate": "2023-05-24T23:50:25+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::12345678012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        }
    }
```

```
}
```

Pour plus d'informations, consultez la section Modification de la durée maximale de session (AWS API) d'un rôle dans le guide de l'utilisateur AWS IAM.

Exemple 3 : pour créer un rôle IAM avec des balises

La commande suivante crée un rôle IAM `Test-Role` avec des balises. Cet exemple utilise l'indicateur de `--tags` paramètre avec les JSON-formatted balises suivantes :`'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. L'indicateur `--tags` peut également être utilisé avec des balises au format raccourci : `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`.

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Sortie :

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role",
        "CreateDate": "2023-05-25T23:29:41+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        },
        "Tags": [
```

```
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

Pour plus d'informations, consultez Étiquette de rôles IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
```

```
type RoleWrapper struct {
 IamClient *iam.Client
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(ctx context.Context, roleName string,
 trustedUserArn string) (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:    "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action:    []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(ctx, &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS SDK pour Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;
import software.amazon.awssdk.services.iam.model.CreateRoleRequest;
import software.amazon.awssdk.services.iam.model.CreateRoleResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import java.io.FileReader;

/*
*   This example requires a trust policy document. For more information, see:
*   https://aws.amazon.com/blogs/security/how-to-use-trust-policies-with-iam-
roles/
*
*
*  In addition, set up your development environment, including your credentials.
*
*  For information, see this documentation topic:
*
*  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class CreateRole {
    public static void main(String[] args) throws Exception {
        final String usage = """
                Usage:
                    <rolename> <fileLocation>\s
```

```
                Where:
                    rolename - The name of the role to create.\s
                    fileLocation - The location of the JSON document that
represents the trust policy.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String rolename = args[0];
        String fileLocation = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMRole(iam, rolename, fileLocation);
        System.out.println("Successfully created user: " + result);
        iam.close();
    }

    public static String createIAMRole(IamClient iam, String rolename, String
fileLocation) throws Exception {
        try {
            JSONObject jsonObject = (JSONObject)
readJsonSimpleDemo(fileLocation);
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(jsonObject.toJSONString())
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
```

```
    }

    public static Object readJsonSimpleDemo(String filename) throws Exception {
        FileReader reader = new FileReader(filename);
        JSONParser jsonParser = new JSONParser();
        return jsonParser.parse(reader);
    }
}
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez le rôle.

```
import { CreateRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const createRole = (roleName) => {
  const command = new CreateRoleCommand({
    AssumeRolePolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Principal: {
            Service: "lambda.amazonaws.com",
```

```
          },
          Action: "sts:AssumeRole",
        },
      ],
    }),
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

    /**
```

```
     * @param string $roleName
     * @param string $rolePolicyDocument
     * @return array
     * @throws AwsException
     */
    public function createRole(string $roleName, string $rolePolicyDocument)
    {
        $result = $this->customWaiter(function () use ($roleName,
 $rolePolicyDocument) {
            return $this->iamClient->createRole([
                'AssumeRolePolicyDocument' => $rolePolicyDocument,
                'RoleName' => $roleName,
            ]);
        });
        return $result['Role'];
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS SDK pour PHP API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple crée un rôle nommé **MyNewRole** et y attache la politique trouvée dans le fichier **NewRoleTrustPolicy.json**. Notez que vous devez utiliser le paramètre booléen **-Raw** pour traiter correctement le fichier de politique JSON. Le document de politique affiché dans la sortie est codé en URL. Il est décodé dans cet exemple à l'aide de la méthode .NET **UrlDecode**.

```
$results = New-IAMRole -AssumeRolePolicyDocument (Get-Content -raw
 NewRoleTrustPolicy.json) -RoleName MyNewRole
$results
```

Sortie :

```
Arn                      : arn:aws:iam::123456789012:role/MyNewRole
AssumeRolePolicyDocument : %7B%0D%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C
%0D%0A%20%20%22Statement%22
```

```
                            %3A%20%5B%0D%0A%20%20%20%20%7B%0D%0A
%20%20%20%20%20%20%22Sid%22%3A%20%22%22%2C
                            %0D%0A%20%20%20%20%20%20%22Effect%22%3A%20%22Allow
%22%2C%0D%0A%20%20%20%20%20%20
                            %22Principal%22%3A%20%7B%0D%0A
%20%20%20%20%20%20%20%20%22AWS%22%3A%20%22arn%3Aaws
                            %3Aiam%3A%3A123456789012%3ADavid%22%0D%0A
%20%20%20%20%20%20%7D%2C%0D%0A%20%20%20
                            %20%20%20%22Action%22%3A%20%22sts%3AAssumeRole%22%0D
%0A%20%20%20%20%7D%0D%0A%20
                            %20%5D%0D%0A%7D
CreateDate               : 4/15/2015 11:04:23 AM
Path                     : /
RoleId                   : V5PAJI2KPN4EAEXAMPLE1
RoleName                 : MyNewRole

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:David"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple crée un rôle nommé **`MyNewRole`** et y attache la politique trouvée dans le fichier **`NewRoleTrustPolicy.json`**. Notez que vous devez utiliser le paramètre booléen **`-Raw`** pour traiter correctement le fichier de politique JSON. Le document de politique affiché dans la sortie est codé en URL. Il est décodé dans cet exemple à l'aide de la méthode .NET **`UrlDecode`**.

```
$results = New-IAMRole -AssumeRolePolicyDocument (Get-Content -raw
 NewRoleTrustPolicy.json) -RoleName MyNewRole
$results
```

Sortie :

```
Arn                       : arn:aws:iam::123456789012:role/MyNewRole
AssumeRolePolicyDocument : %7B%0D%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C
%0D%0A%20%20%22Statement%22
                           %3A%20%5B%0D%0A%20%20%20%20%7B%0D%0A
%20%20%20%20%20%20%22Sid%22%3A%20%22%22%2C
                           %0D%0A%20%20%20%20%20%20%22Effect%22%3A%20%22Allow
%22%2C%0D%0A%20%20%20%20%20%20
                           %22Principal%22%3A%20%7B%0D%0A
%20%20%20%20%20%20%20%20%22AWS%22%3A%20%22arn%3Aaws
                           %3Aiam%3A%3A123456789012%3ADavid%22%0D%0A
%20%20%20%20%20%20%7D%2C%0D%0A%20%20%20
                           %20%20%20%22Action%22%3A%20%22sts%3AAssumeRole%22%0D
%0A%20%20%20%20%7D%0D%0A%20
                           %20%5D%0D%0A%7D
CreateDate                : 4/15/2015 11:04:23 AM
Path                      : /
RoleId                    : V5PAJI2KPN4EAEXAMPLE1
RoleName                  : MyNewRole

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:David"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
    :return: The newly created role.
    """
    trust_policy = {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
            }
            for service in allowed_services
        ],
    }

    try:
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
```

```
    else:
        return role
```

- Pour plus de détails sur l'API, consultez CreateRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Creates a role and attaches policies to it.
 #
 # @param role_name [String] The name of the role.
 # @param assume_role_policy_document [Hash] The trust relationship policy
document.
 # @param policy_arns [Array<String>] The ARNs of the policies to attach.
 # @return [String, nil] The ARN of the new role if successful, or nil if an
error occurred.
 def create_role(role_name, assume_role_policy_document, policy_arns)
   response = @iam_client.create_role(
     role_name: role_name,
     assume_role_policy_document: assume_role_policy_document.to_json
   )
   role_arn = response.role.arn

   policy_arns.each do |policy_arn|
     @iam_client.attach_role_policy(
       role_name: role_name,
       policy_arn: policy_arn
     )
   end
```

```
    role_arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating role: #{e.message}")
    nil
  end
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn create_role(
    client: &iamClient,
    role_name: &str,
    role_policy_document: &str,
) -> Result<Role, iamError> {
    let response: CreateRoleOutput = loop {
        if let Ok(response) = client
            .create_role()
            .role_name(role_name)
            .assume_role_policy_document(role_policy_document)
            .send()
            .await
        {
            break response;
        }
    };

    Ok(response.role.unwrap())
}
```

- Pour plus de détails sur l'API, voir CreateRolela section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    oo_result = lo_iam->createrole(
      iv_rolename = iv_role_name
      iv_assumerolepolicydocument = iv_assume_role_policy_document ).
    MESSAGE 'Role created successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamentityalrdyexex.
    MESSAGE 'Role already exists.' TYPE 'E'.
  CATCH /aws1/cx_iammalformedplydocex.
    MESSAGE 'Assume role policy document is malformed.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Role limit exceeded.' TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func createRole(name: String, policyDocument: String) async throws ->
 String {
        let input = CreateRoleInput(
            assumeRolePolicyDocument: policyDocument,
            roleName: name
        )
        do {
            let output = try await client.createRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
            guard let id = role.roleId else {
                throw ServiceHandlerError.noSuchRole
            }
            return id
        } catch {
            print("ERROR: createRole:", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisez **`CreateSAMLProvider`** avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `CreateSAMLProvider`.

CLI

AWS CLI

Pour créer un fournisseur SAML

Cet exemple crée un nouveau fournisseur SAML dans IAM nommé `MySAMLProvider`. Il est décrit par le document de métadonnées SAML présent dans le fichier `SAMLMetaData.xml`.

```
aws iam create-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --name MySAMLProvider
```

Sortie :

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/MySAMLProvider"
}
```

Pour plus d'informations, consultez Création de fournisseurs d'identité SAML IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus d'informations sur l'API, consultez la rubrique CreateSAMLProvider dans la Référence des commandes AWS CLI .

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { CreateSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "node:fs";
import * as path from "node:path";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";

const client = new IAMClient({});

/**
 * This sample document was generated using Auth0.
 * For more information on generating this document,
 see https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_providers_create_saml.html#samlstep1.
```

```
 */
const sampleMetadataDocument = readFileSync(
  path.join(
    dirnameFromMetaUrl(import.meta.url),
    "../../../../resources/sample_files/sample_saml_metadata.xml",
  ),
);

/**
 *
 * @param {*} providerName
 * @returns
 */
export const createSAMLProvider = async (providerName) => {
  const command = new CreateSAMLProviderCommand({
    Name: providerName,
    SAMLMetadataDocument: sampleMetadataDocument.toString(),
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Pour les détails de l'API, consultez CreateSAMLProvider dans la Référence de l'API AWS SDK pour JavaScript .

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple crée une entité de fournisseur SAML dans IAM. Il est nommé **`MySAMLProvider`** et est décrit par le document de métadonnées SAML trouvé dans le fichier **`SAMLMetaData.xml`**, qui a été téléchargé séparément depuis le site Web du fournisseur de services SAML.

```
New-IAMSAMLProvider -Name MySAMLProvider -SAMLMetadataDocument (Get-Content -Raw
 SAMLMetaData.xml)
```

Sortie :

```
arn:aws:iam::123456789012:saml-provider/MySAMLProvider
```

- Pour plus de détails sur l'API, consultez CreateSAMLProvider dans la Référence des applets de commande pour les Outils AWS pour PowerShell (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple crée une entité de fournisseur SAML dans IAM. Il est nommé **`MySAMLProvider`** et est décrit par le document de métadonnées SAML trouvé dans le fichier **`SAMLMetaData.xml`**, qui a été téléchargé séparément depuis le site Web du fournisseur de services SAML.

```
New-IAMSAMLProvider -Name MySAMLProvider -SAMLMetadataDocument (Get-Content -Raw
 SAMLMetaData.xml)
```

Sortie :

```
arn:aws:iam::123456789012:saml-provider/MySAMLProvider
```

- Pour plus de détails sur l'API, consultez CreateSAMLProvider dans la Référence des applets de commande pour les Outils AWS pour PowerShell (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`CreateServiceLinkedRole`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `CreateServiceLinkedRole`.

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour créer un rôle lié à un service

L'`create-service-linked-role`exemple suivant crée un rôle lié à un service pour le AWS service spécifié et y joint la description spécifiée.

```
aws iam create-service-linked-role \
    --aws-service-name lex.amazonaws.com \
    --description "My service-linked role to support Lex"
```

Sortie :

```
{
    "Role": {
```

```
        "Path": "/aws-service-role/lex.amazonaws.com/",
        "RoleName": "AWSServiceRoleForLexBots",
        "RoleId": "AROA1234567890EXAMPLE",
        "Arn": "arn:aws:iam::1234567890:role/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots",
        "CreateDate": "2019-04-17T20:34:14+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Action": [
                        "sts:AssumeRole"
                    ],
                    "Effect": "Allow",
                    "Principal": {
                        "Service": [
                            "lex.amazonaws.com"
                        ]
                    }
                }
            ]
        }
    }
}
```

Pour plus d'informations, consultez Utilisation des rôles liés à un service dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(ctx context.Context,
 serviceName string, description string) (
 *types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(ctx,
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section Référence des AWS SDK pour Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez un rôle lié à un service.

```
import {
  CreateServiceLinkedRoleCommand,
  GetRoleCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} serviceName
 */
export const createServiceLinkedRole = async (serviceName) => {
  const command = new CreateServiceLinkedRoleCommand({
    // For a list of AWS services that support service-linked roles,
    // see https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_aws-
services-that-work-with-iam.html.
    //
    // For a list of AWS service endpoints, see https://docs.aws.amazon.com/
general/latest/gr/aws-service-information.html.
    AWSServiceName: serviceName,
  });
  try {
    const response = await client.send(command);
    console.log(response);
    return response;
  } catch (caught) {
    if (
      caught instanceof Error &&
      caught.name === "InvalidInputException" &&
      caught.message.includes(
```

```
        "Service role name AWSServiceRoleForElasticBeanstalk has been taken in
 this account",
      )
    ) {
      console.warn(caught.message);
      return client.send(
        new GetRoleCommand({ RoleName: "AWSServiceRoleForElasticBeanstalk" }),
      );
    }
    throw caught;
  }
};
```

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function createServiceLinkedRole($awsServiceName, $customSuffix = "",
 $description = "")
    {
        $createServiceLinkedRoleArguments = ['AWSServiceName' =>
 $awsServiceName];
        if ($customSuffix) {
            $createServiceLinkedRoleArguments['CustomSuffix'] = $customSuffix;
        }
        if ($description) {
            $createServiceLinkedRoleArguments['Description'] = $description;
        }
```

```
        return $this->iamClient-
>createServiceLinkedRole($createServiceLinkedRoleArguments);
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section Référence des AWS SDK pour PHP API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple crée un rôle lié à un service pour le service de mise à l'échelle automatique.

```
New-IAMServiceLinkedRole -AWSServiceName autoscaling.amazonaws.com -CustomSuffix
 RoleNameEndsWithThis -Description "My service-linked role to support
 autoscaling"
```

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple crée un rôle lié à un service pour le service de mise à l'échelle automatique.

```
New-IAMServiceLinkedRole -AWSServiceName autoscaling.amazonaws.com -CustomSuffix
 RoleNameEndsWithThis -Description "My service-linked role to support
 autoscaling"
```

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def create_service_linked_role(service_name, description):
    """
    Creates a service-linked role.

    :param service_name: The name of the service that owns the role.
    :param description: A description to give the role.
    :return: The newly created role.
    """
    try:
        response = iam.meta.client.create_service_linked_role(
            AWSServiceName=service_name, Description=description
        )
        role = iam.Role(response["Role"]["RoleName"])
        logger.info("Created service-linked role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create service-linked role for %s.",
 service_name)
        raise
    else:
        return role
```

- Pour plus de détails sur l'API, consultez CreateServiceLinkedRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Creates a service-linked role
 #
 # @param service_name [String] The service name to create the role for.
 # @param description [String] The description of the service-linked role.
 # @param suffix [String] Suffix for customizing role name.
 # @return [String] The name of the created role
 def create_service_linked_role(service_name, description, suffix)
   response = @iam_client.create_service_linked_role(
     aws_service_name: service_name, description: description, custom_suffix:
suffix
   )
   role_name = response.role.role_name
   @logger.info("Created service-linked role #{role_name}.")
   role_name
 rescue Aws::Errors::ServiceError => e
   @logger.error("Couldn't create service-linked role for #{service_name}.
Here's why:")
   @logger.error("\t#{e.code}: #{e.message}")
   raise
 end
```

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn create_service_linked_role(
    client: &iamClient,
    aws_service_name: String,
    custom_suffix: Option<String>,
    description: Option<String>,
) -> Result<CreateServiceLinkedRoleOutput,
 SdkError<CreateServiceLinkedRoleError>> {
    let response = client
        .create_service_linked_role()
        .aws_service_name(aws_service_name)
        .set_custom_suffix(custom_suffix)
        .set_description(description)
        .send()
        .await?;

    Ok(response)
}
```

- Pour plus de détails sur l'API, voir CreateServiceLinkedRolela section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->listpolicyversions(
          iv_policyarn = iv_policy_arn ).
        MESSAGE 'Retrieved policy versions list.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing policy versions.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func createServiceLinkedRole(service: String, suffix: String? = nil,
 description: String?)
                    async throws -> IAMClientTypes.Role {
        let input = CreateServiceLinkedRoleInput(
            awsServiceName: service,
            customSuffix: suffix,
            description: description
        )
        do {
            let output = try await client.createServiceLinkedRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
```

```
            return role
        } catch {
            print("ERROR: createServiceLinkedRole:", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`CreateUser`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `CreateUser`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Principes de base
- Créer des utilisateurs en lecture seule et en lecture-écriture

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
/// <summary>
/// Create an IAM user.
/// </summary>
/// <param name="userName">The username for the new IAM user.</param>
```

```
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
 { UserName = userName });
        return response.User;
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK pour .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
#############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
```

```
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
########################################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
```

```
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreateUserRequest create_request;
    create_request.SetUserName(userName);

    auto create_outcome = iam.CreateUser(create_request);
    if (!create_outcome.IsSuccess()) {
        std::cerr << "Error creating IAM user " << userName << ":" <<
                  create_outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully created IAM user " << userName << std::endl;
    }

    return create_outcome.IsSuccess();
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Exemple 1 : pour créer un utilisateur IAM

La commande `create-user` suivante crée un utilisateur IAM nommé Bob dans le compte actuel.

```
aws iam create-user \
    --user-name Bob
```

Sortie :

```
{
    "User": {
        "UserName": "Bob",
        "Path": "/",
        "CreateDate": "2023-06-08T03:20:41.270Z",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Bob"
    }
}
```

Pour plus d'informations, consultez la section Création d'un utilisateur IAM dans votre AWS compte dans le guide de l'utilisateur AWS IAM.

Exemple 2 : pour créer un utilisateur IAM à un chemin spécifié

La commande `create-user` suivante crée un utilisateur IAM nommé Bob au chemin spécifié.

```
aws iam create-user \
    --user-name Bob \
    --path /division_abc/subdivision_xyz/
```

Sortie :

```
{
    "User": {
        "Path": "/division_abc/subdivision_xyz/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/division_abc/subdivision_xyz/Bob",
        "CreateDate": "2023-05-24T18:20:17+00:00"
    }
}
```

Pour plus d'informations, consultez Identifiants IAM dans le Guide de l'utilisateur AWS IAM.

Exemple 3 : pour créer un utilisateur IAM avec des balises

La commande `create-user` suivante crée un utilisateur IAM nommé Bob avec des balises. Cet exemple utilise l'indicateur de `--tags` paramètre avec les JSON-formatted balises suivantes :`'{"Key": "Department", "Value": "Accounting"}' '{"Key":`

`"Location", "Value": "Seattle"}'`. L'indicateur `--tags` peut également être utilisé avec des balises au format raccourci : `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`.

```
aws iam create-user \
    --user-name Bob \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Sortie :

```
{
    "User": {
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-25T17:14:21+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

Pour plus d'informations, consultez Étiquette d'utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

Exemple 3 : pour créer un utilisateur IAM avec une limite des autorisations définie

La `create-user` commande suivante crée un utilisateur IAM nommé Bob avec la limite d'autorisations de AmazonS3FullAccess.

```
aws iam create-user \
    --user-name Bob \
    --permissions-boundary arn:aws:iam::aws:policy/AmazonS3FullAccess
```

Sortie :

```
{
    "User": {
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-24T17:50:53+00:00",
        "PermissionsBoundary": {
        "PermissionsBoundaryType": "Policy",
        "PermissionsBoundaryArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
        }
    }
}
```

Pour plus d'informations, consultez Limites d'autorisations pour les entités IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
```

```
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(ctx, &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK pour Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreateUserRequest;
import software.amazon.awssdk.services.iam.model.CreateUserResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.waiters.IamWaiter;
import software.amazon.awssdk.services.iam.model.GetUserRequest;
import software.amazon.awssdk.services.iam.model.GetUserResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username>\s

                Where:
                    username - The name of the user to create.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMUser(iam, username);
        System.out.println("Successfully created user: " + result);
```

```
        iam.close();
    }

    public static String createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            CreateUserResponse response = iam.createUser(request);

            // Wait until the user is created.
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
 iamWaiter.waitUntilUserExists(userRequest);

 waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user().userName();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez l'utilisateur.

```
import { CreateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} name
 */
export const createUser = (name) => {
  const command = new CreateUserCommand({ UserName: name });
  return client.send(command);
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
```

```
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    iam.createUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
      } else {
        console.log("Success", data);
      }
    });
  } else {
    console.log(
      "User " + process.argv[2] + " already exists",
      data.User.UserId
    );
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun createIAMUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}
```

- Pour plus de détails sur l'API, consultez CreateUserla section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";


    /**
     * @param string $name
     * @return array
     * @throws AwsException
     */
    public function createUser(string $name): array
    {
        $result = $this->iamClient->createUser([
```

```
            'UserName' => $name,
        ]);

        return $result['User'];
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK pour PHP API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple crée un utilisateur IAM nommé **Bob**. Si Bob doit se connecter à la AWS console, vous devez exécuter la commande séparément **New-IAMLoginProfile** pour créer un profil de connexion avec un mot de passe. Si Bob doit exécuter AWS PowerShell des commandes CLI multiplateformes ou effectuer des appels d' AWS API, vous devez exécuter la **New-IAMAccessKey** commande séparément pour créer des clés d'accès.

```
New-IAMUser -UserName Bob
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 4/22/2015 12:02:11 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : AIDAJWGEFDMEMEXAMPLE1
UserName         : Bob
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple crée un utilisateur IAM nommé **Bob**. Si Bob doit se connecter à la AWS console, vous devez exécuter la commande séparément **New-IAMLoginProfile** pour créer un profil de connexion avec un mot de passe. Si Bob doit exécuter AWS PowerShell des commandes CLI multiplateformes ou effectuer des appels d' AWS API, vous devez exécuter la **New-IAMAccessKey** commande séparément pour créer des clés d'accès.

```
New-IAMUser -UserName Bob
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 4/22/2015 12:02:11 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : AIDAJWGEFDMEMEXAMPLE1
UserName         : Bob
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def create_user(user_name):
    """
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
        logger.exception("Couldn't create user %s.", user_name)
        raise
    else:
        return user
```

- Pour plus de détails sur l'API, consultez CreateUserle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Creates a user and their login profile
 #
 # @param user_name [String] The name of the user
 # @param initial_password [String] The initial password for the user
 # @return [String, nil] The ID of the user if created, or nil if an error
occurred
 def create_user(user_name, initial_password)
   response = @iam_client.create_user(user_name: user_name)
   @iam_client.wait_until(:user_exists, user_name: user_name)
   @iam_client.create_login_profile(
     user_name: user_name,
     password: initial_password,
     password_reset_required: true
   )
   @logger.info("User '#{user_name}' created successfully.")
   response.user.user_id
 rescue Aws::IAM::Errors::EntityAlreadyExists
   @logger.error("Error creating user '#{user_name}': user already exists.")
   nil
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error creating user '#{user_name}': #{e.message}")
   nil
 end
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn create_user(client: &iamClient, user_name: &str) -> Result<User,
 iamError> {
    let response = client.create_user().user_name(user_name).send().await?;

    Ok(response.user.unwrap())
}
```

- Pour plus de détails sur l'API, voir CreateUserla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->createuser(
          iv_username = iv_user_name ).
        MESSAGE 'User created successfully.' TYPE 'I'.
```

```
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'User already exists.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Limit exceeded for IAM users.' TYPE 'E'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Entity does not exist.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func createUser(name: String) async throws -> String {
        let input = CreateUserInput(
            userName: name
        )
        do {
            let output = try await client.createUser(input: input)
            guard let user = output.user else {
                throw ServiceHandlerError.noSuchUser
            }
            guard let id = user.userId else {
                throw ServiceHandlerError.noSuchUser
            }
            return id
        } catch {
            print("ERROR: createUser:", dump(error))
```

```
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`CreateVirtualMfaDevice`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `CreateVirtualMfaDevice`.

CLI

AWS CLI

Pour créer un dispositif MFA virtuel

Cet exemple crée un dispositif MFA virtuel appelé `BobsMFADevice`. Il crée un fichier, appelé `QRCode.png`, qui contient les informations d'amorçage et le place dans le répertoire `C:/`. La méthode d'amorçage utilisée dans cet exemple est `QRCodePNG`.

```
aws iam create-virtual-mfa-device \
    --virtual-mfa-device-name BobsMFADevice \
    --outfile C:/QRCode.png \
    --bootstrap-method QRCodePNG
```

Sortie :

```
{
    "VirtualMFADevice": {
        "SerialNumber": "arn:aws:iam::210987654321:mfa/BobsMFADevice"
}
```

Pour plus d'informations, consultez Utilisation de l'authentification multifactorielle (MFA) dans AWS dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous CreateVirtualMfaDeviceà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple crée un dispositif MFA virtuel. Les lignes 2 et 3 extraient la valeur **`Base32StringSeed`** dont le logiciel MFA virtuel a besoin pour créer un compte (comme alternative au code QR). Après avoir configuré le programme avec la valeur, obtenez deux codes d'authentification séquentiels auprès du programme. Enfin, utilisez la dernière commande pour associer le dispositif MFA virtuel à l'utilisateur IAM **Bob** et synchroniser le compte avec les deux codes d'authentification.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$SR = New-Object System.IO.StreamReader($Device.Base32StringSeed)
$base32stringseed = $SR.ReadToEnd()
$base32stringseed
CZWZMCQNW4DEXAMPLE3VOUGXJFZYSUW7EXAMPLECR4NJFD65GX2SLUDW2EXAMPLE
```

Sortie :

```
-- Pause here to enter base-32 string seed code into virtual MFA program to
 register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

Exemple 2 : cet exemple crée un dispositif MFA virtuel. Les lignes 2 et 3 extraient la valeur **`QRCodePNG`** et l'écrivent dans un fichier. Cette image peut être numérisée par le logiciel MFA virtuel pour créer un compte (au lieu de saisir manuellement la Base32StringSeed valeur). Après avoir créé le compte dans votre programme MFA virtuel, obtenez deux codes d'authentification séquentiels et saisissez-les dans les dernières commandes pour associer le dispositif MFA virtuel à l'utilisateur IAM **Bob** et synchroniser le compte

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$BR = New-Object System.IO.BinaryReader($Device.QRCodePNG)
$BR.ReadBytes($BR.BaseStream.Length) | Set-Content -Encoding Byte -Path
 QRCode.png
```

Sortie :

```
 -- Pause here to scan PNG with virtual MFA program to register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

- Pour plus de détails sur l'API, reportez-vous CreateVirtualMfaDeviceà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple crée un dispositif MFA virtuel. Les lignes 2 et 3 extraient la valeur **`Base32StringSeed`** dont le logiciel MFA virtuel a besoin pour créer un compte (comme alternative au code QR). Après avoir configuré le programme avec la valeur, obtenez deux codes d'authentification séquentiels auprès du programme. Enfin, utilisez la dernière commande pour associer le dispositif MFA virtuel à l'utilisateur IAM **`Bob`** et synchroniser le compte avec les deux codes d'authentification.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$SR = New-Object System.IO.StreamReader($Device.Base32StringSeed)
$base32stringseed = $SR.ReadToEnd()
$base32stringseed
CZWZMCQNW4DEXAMPLE3VOUGXJFZYSUW7EXAMPLECR4NJFD65GX2SLUDW2EXAMPLE
```

Sortie :

```
-- Pause here to enter base-32 string seed code into virtual MFA program to
 register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

Exemple 2 : cet exemple crée un dispositif MFA virtuel. Les lignes 2 et 3 extraient la valeur **`QRCodePNG`** et l'écrivent dans un fichier. Cette image peut être numérisée par le logiciel MFA virtuel pour créer un compte (au lieu de saisir manuellement la Base32StringSeed valeur). Après avoir créé le compte dans votre programme MFA virtuel, obtenez deux codes d'authentification séquentiels et saisissez-les dans les dernières commandes pour associer le dispositif MFA virtuel à l'utilisateur IAM **`Bob`** et synchroniser le compte

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
```

```
$BR = New-Object System.IO.BinaryReader($Device.QRCodePNG)
$BR.ReadBytes($BR.BaseStream.Length) | Set-Content -Encoding Byte -Path
 QRCode.png
```

Sortie :

```
 -- Pause here to scan PNG with virtual MFA program to register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

- Pour plus de détails sur l'API, reportez-vous CreateVirtualMfaDeviceà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`DeactivateMfaDevice`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `DeactivateMfaDevice`.

CLI

AWS CLI

Pour désactiver un dispositif MFA

Cette commande désactive le dispositif MFA virtuel, dont l'ARN est `arn:aws:iam::210987654321:mfa/BobsMFADevice`, associé à l'utilisateur Bob.

```
aws iam deactivate-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Utilisation de l'authentification multifactorielle (MFA) dans AWS dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeactivateMfaDeviceà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cette commande désactive le dispositif MFA matériel associé à l'utilisateur **Bob** et possédant le numéro de série **123456789012**.

```
Disable-IAMMFADevice -UserName "Bob" -SerialNumber "123456789012"
```

Exemple 2 : cette commande désactive le dispositif MFA virtuel associé à l'utilisateur **David** qui possède l'ARN **arn:aws:iam::210987654321:mfa/David**. Notez que le dispositif MFA virtuel n'est pas supprimé du compte. Le dispositif virtuel est toujours présent et apparaît dans la sortie de la commande **Get-IAMVirtualMFADevice**. Avant de pouvoir créer un dispositif MFA virtuel pour le même utilisateur, vous devez supprimer l'ancien à l'aide de la **Remove-IAMVirtualMFADevice** commande.

```
Disable-IAMMFADevice -UserName "David" -SerialNumber
 "arn:aws:iam::210987654321:mfa/David"
```

- Pour plus de détails sur l'API, reportez-vous DeactivateMfaDeviceà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cette commande désactive le dispositif MFA matériel associé à l'utilisateur **Bob** et possédant le numéro de série **123456789012**.

```
Disable-IAMMFADevice -UserName "Bob" -SerialNumber "123456789012"
```

Exemple 2 : cette commande désactive le dispositif MFA virtuel associé à l'utilisateur **David** qui possède l'ARN **arn:aws:iam::210987654321:mfa/David**. Notez que le dispositif MFA virtuel n'est pas supprimé du compte. Le dispositif virtuel est toujours présent et apparaît dans la sortie de la commande **Get-IAMVirtualMFADevice**. Avant de pouvoir créer un dispositif MFA virtuel pour le même utilisateur, vous devez supprimer l'ancien à l'aide de la **Remove-IAMVirtualMFADevice** commande.

```
Disable-IAMMFADevice -UserName "David" -SerialNumber
 "arn:aws:iam::210987654321:mfa/David"
```

- Pour plus de détails sur l'API, reportez-vous DeactivateMfaDeviceà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`DeleteAccessKey`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `DeleteAccessKey`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Principes de base
- Créer des utilisateurs en lecture seule et en lecture-écriture
- Gérer des clés d'accès

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
```

```
        var response = await _IAMService.DeleteAccessKeyAsync(new
 DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccessKeyà la section Référence des AWS SDK pour .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
```

```
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an AWS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi
```

```
  if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccessKeyà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::deleteAccessKey(const Aws::String &userName,
                                  const Aws::String &accessKeyID,
                                  const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccessKeyRequest request;
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);

    auto outcome = iam.DeleteAccessKey(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting access key " << accessKeyID << " from user "
                  << userName << ": " << outcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully deleted access key " << accessKeyID
                  << " for IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccessKeyà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour supprimer une clé d'accès d'un utilisateur IAM

La commande `delete-access-key` suivante supprime la clé d'accès spécifiée (un ID de clé d'accès rapide et une clé d'accès secrète) de l'utilisateur IAM nommé Bob.

```
aws iam delete-access-key \
    --access-key-id AKIDPMS9RO4H3FEXAMPLE \
    --user-name Bob
```

Cette commande ne produit aucune sortie.

Pour répertorier les clés d'accès définies pour un utilisateur IAM, utilisez la commande `list-access-keys`.

Pour plus d'informations, consultez Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteAccessKeyà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}
```

```
// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(ctx context.Context, userName string,
 keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(ctx, &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccessKeyà la section Référence des AWS SDK pour Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteAccessKeyRequest;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
```

```
 */
public class DeleteAccessKey {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username> <accessKey>\s

                Where:
                    username - The name of the user.\s
                    accessKey - The access key ID for the secret access key you
 want to delete.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        String accessKey = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        deleteKey(iam, username, accessKey);
        iam.close();
    }

    public static void deleteKey(IamClient iam, String username, String
 accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
```

```
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccessKeyà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Supprimez la clé d'accès.

```
import { DeleteAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
 */
export const deleteAccessKey = (userName, accessKeyId) => {
  const command = new DeleteAccessKeyCommand({
    AccessKeyId: accessKeyId,
    UserName: userName,
  });

  return client.send(command);
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.

- Pour plus de détails sur l'API, reportez-vous DeleteAccessKeyà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  AccessKeyId: "ACCESS_KEY_ID",
  UserName: "USER_NAME",
};

iam.deleteAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous DeleteAccessKeyà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun deleteKey(
    userNameVal: String,
    accessKey: String,
) {
    val request =
        DeleteAccessKeyRequest {
            accessKeyId = accessKey
            userName = userNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccessKey(request)
        println("Successfully deleted access key $accessKey from $userNameVal")
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccessKeyà la section AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple supprime la paire de clés AWS d'accès avec l'ID **AKIAIOSFODNN7EXAMPLE** de clé de l'utilisateur nommé**Bob**.

```
Remove-IAMAccessKey -AccessKeyId AKIAIOSFODNN7EXAMPLE -UserName Bob -Force
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccessKeyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple supprime la paire de clés AWS d'accès avec l'ID **AKIAIOSFODNN7EXAMPLE** de clé de l'utilisateur nommé**Bob**.

```
Remove-IAMAccessKey -AccessKeyId AKIAIOSFODNN7EXAMPLE -UserName Bob -Force
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccessKeyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

- Pour plus de détails sur l'API, consultez DeleteAccessKeyle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, désactive et supprime les clés d'accès.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'AccessKeyManager'
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
```

```
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded
    @logger.error('Error creating access key: limit exceeded. Cannot create
more.')
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: 'Inactive'
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
```

```
    false
  end
end
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccessKeyà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn delete_access_key(
    client: &iamClient,
    user: &User,
    key: &AccessKey,
) -> Result<(), iamError> {
    loop {
        match client
            .delete_access_key()
            .user_name(user.user_name())
            .access_key_id(key.access_key_id())
            .send()
            .await
        {
            Ok(_) => {
                break;
            }
            Err(e) => {
                println!("Can't delete the access key: {:?}", e);
                sleep(Duration::from_secs(2)).await;
            }
        }
    }
    Ok(())
```

```
}
```

- Pour plus de détails sur l'API, voir DeleteAccessKeyla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    lo_iam->deleteaccesskey(
      iv_accesskeyid = iv_access_key_id
      iv_username = iv_user_name ).
    MESSAGE 'Access key deleted successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Access key or user does not exist.' TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccessKeyà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func deleteAccessKey(user: IAMClientTypes.User? = nil,
                                key: IAMClientTypes.AccessKey) async throws
    {
        let userName: String?

        if user != nil {
            userName = user!.userName
        } else {
            userName = nil
        }

        let input = DeleteAccessKeyInput(
            accessKeyId: key.accessKeyId,
            userName: userName
        )
        do {
            _ = try await iamClient.deleteAccessKey(input: input)
        } catch {
            print("ERROR: deleteAccessKey:", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccessKeyà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`DeleteAccountAlias`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `DeleteAccountAlias`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer votre compte

C++

SDK pour C++

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::deleteAccountAlias(const Aws::String &accountAlias,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccountAliasRequest request;
    request.SetAccountAlias(accountAlias);

    const auto outcome = iam.DeleteAccountAlias(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting account alias " << accountAlias << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted account alias " << accountAlias <<
                  std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccountAliasà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour supprimer un alias de compte

La commande `delete-account-alias` suivante supprime les alias `mycompany` du compte actuel.

```
aws iam delete-account-alias \
    --account-alias mycompany
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez la section Votre identifiant de AWS compte et son alias dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteAccountAliasà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.DeleteAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
```

```
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteAccountAlias {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <alias>\s

                Where:
                    alias - The account alias to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMAccountAlias(iam, alias);
        iam.close();
    }

    public static void deleteIAMAccountAlias(IamClient iam, String alias) {
        try {
            DeleteAccountAliasRequest request =
 DeleteAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();

            iam.deleteAccountAlias(request);
            System.out.println("Successfully deleted account alias " + alias);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
```

```
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccountAliasà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Supprimez l'alias de compte.

```
import { DeleteAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} alias
 */
export const deleteAccountAlias = (alias) => {
  const command = new DeleteAccountAliasCommand({ AccountAlias: alias });

  return client.send(command);
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous DeleteAccountAliasà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous DeleteAccountAliasà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun deleteIAMAccountAlias(alias: String) {
    val request =
        DeleteAccountAliasRequest {
            accountAlias = alias
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccountAlias(request)
        println("Successfully deleted account alias $alias")
    }
}
```

- Pour plus de détails sur l'API, consultez DeleteAccountAliasla section AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple supprime l'alias de compte de votre Compte AWS. La page de connexion de l'utilisateur avec l'alias at https://mycompanyaws.signin.aws.amazon.com/console ne fonctionne plus. Vous devez plutôt utiliser l'URL d'origine avec votre numéro d'Compte AWS identification sur https ://<accountidnumber>.signin.aws.amazon. com/console.

```
Remove-IAMAccountAlias -AccountAlias mycompanyaws
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccountAliasà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple supprime l'alias de compte de votre Compte AWS. La page de connexion de l'utilisateur avec l'alias at https://mycompanyaws.signin.aws.amazon.com/console ne fonctionne plus. Vous devez plutôt utiliser l'URL d'origine avec votre numéro d'Compte AWS identification sur https ://<accountidnumber>.signin.aws.amazon. com/console.

```
Remove-IAMAccountAlias -AccountAlias mycompanyaws
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccountAliasà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise
```

- Pour plus de détails sur l'API, consultez DeleteAccountAliasle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Listez, créez et supprimez les alias de compte.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info('Account aliases are:')
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info('No account aliases found.')
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccountAliasà la section Référence des AWS SDK pour Ruby API.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        lo_iam->deleteaccountalias(
          iv_accountalias = iv_account_alias ).
        MESSAGE 'Account alias deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Account alias does not exist.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, consultez DeleteAccountAliasla section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`DeleteAccountPasswordPolicy`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `DeleteAccountPasswordPolicy`.

CLI

AWS CLI

Pour supprimer la politique de mot de passe du compte actuel

La commande `delete-account-password-policy` suivante supprime la politique de mot de passe du compte actuel.

```
aws iam delete-account-password-policy
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Définition d'une politique de mot de passe du compte pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteAccountPasswordPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple supprime la politique de mot de passe pour le Compte AWS et rétablit toutes les valeurs par défaut d'origine. Si aucune politique de mot de passe n'existe actuellement, le message d'erreur suivant s'affiche : PasswordPolicy Impossible de trouver la politique de compte portant le nom.

```
Remove-IAMAccountPasswordPolicy
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccountPasswordPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple supprime la politique de mot de passe pour le Compte AWS et rétablit toutes les valeurs par défaut d'origine. Si aucune politique de mot de passe n'existe actuellement, le message d'erreur suivant s'affiche : PasswordPolicy Impossible de trouver la politique de compte portant le nom.

```
Remove-IAMAccountPasswordPolicy
```

- Pour plus de détails sur l'API, reportez-vous DeleteAccountPasswordPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`DeleteGroup`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `DeleteGroup`.

CLI

AWS CLI

Pour supprimer un groupe IAM

La commande `delete-group` suivante supprime un groupe IAM nommé `MyTestGroup`.

```
aws iam delete-group \
    --group-name MyTestGroup
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Suppression d'un groupe d'utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteGroupà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { DeleteGroupCommand, IAMClient } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

/**
 *
 * @param {string} groupName
 */
export const deleteGroup = async (groupName) => {
  const command = new DeleteGroupCommand({
    GroupName: groupName,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Pour plus de détails sur l'API, reportez-vous DeleteGroupà la section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime le groupe IAM nommé **`MyTestGroup`**. La première commande supprime tous les utilisateurs IAM membres du groupe, et la seconde supprime le groupe IAM. Les deux commandes fonctionnent sans qu'aucune invite de confirmation ne soit requise.

```
(Get-IAMGroup -GroupName MyTestGroup).Users | Remove-IAMUserFromGroup -GroupName
 MyTestGroup -Force
Remove-IAMGroup -GroupName MyTestGroup -Force
```

- Pour plus de détails sur l'API, reportez-vous DeleteGroupà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime le groupe IAM nommé **`MyTestGroup`**. La première commande supprime tous les utilisateurs IAM membres du groupe, et la seconde supprime le

groupe IAM. Les deux commandes fonctionnent sans qu'aucune invite de confirmation ne soit requise.

```
(Get-IAMGroup -GroupName MyTestGroup).Users | Remove-IAMUserFromGroup -GroupName
 MyTestGroup -Force
Remove-IAMGroup -GroupName MyTestGroup -Force
```

- Pour plus de détails sur l'API, reportez-vous DeleteGroupà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`DeleteGroupPolicy`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `DeleteGroupPolicy`.

CLI

AWS CLI

Pour supprimer une politique d'un groupe IAM

La commande `delete-group-policy` suivante supprime la politique nommée `ExamplePolicy` du groupe nommé `Admins`.

```
aws iam delete-group-policy \
    --group-name Admins \
    --policy-name ExamplePolicy
```

Cette commande ne produit aucune sortie.

Pour voir les politiques attachées à un groupe, utilisez la commande `list-group-policies`.

Pour plus d'informations, consultez Gestion des politiques IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteGroupPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime la politique en ligne nommée **`TesterPolicy`** du groupe IAM **`Testers`**. Les utilisateurs de ce groupe perdent immédiatement les autorisations définies dans cette politique.

```
Remove-IAMGroupPolicy -GroupName Testers -PolicyName TestPolicy
```

- Pour plus de détails sur l'API, reportez-vous DeleteGroupPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime la politique en ligne nommée **`TesterPolicy`** du groupe IAM **`Testers`**. Les utilisateurs de ce groupe perdent immédiatement les autorisations définies dans cette politique.

```
Remove-IAMGroupPolicy -GroupName Testers -PolicyName TestPolicy
```

- Pour plus de détails sur l'API, reportez-vous DeleteGroupPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`DeleteInstanceProfile`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `DeleteInstanceProfile`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Créer et gérer un service résilient
- Débuter avec le streaming géré
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /// <summary>
   /// Detaches a role from an instance profile, detaches policies from the
role,
   /// and deletes all the resources.
   /// </summary>
   /// <param name="profileName">The name of the profile to delete.</param>
   /// <param name="roleName">The name of the role to delete.</param>
   /// <returns>Async task.</returns>
   public async Task DeleteInstanceProfile(string profileName, string roleName)
   {
       try
       {
           await _amazonIam.RemoveRoleFromInstanceProfileAsync(
               new RemoveRoleFromInstanceProfileRequest()
               {
                   InstanceProfileName = profileName,
                   RoleName = roleName
               });
           await _amazonIam.DeleteInstanceProfileAsync(
               new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
           var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
               new ListAttachedRolePoliciesRequest() { RoleName = roleName });
           foreach (var policy in attachedPolicies.AttachedPolicies)
           {
               await _amazonIam.DetachRolePolicyAsync(
                   new DetachRolePolicyRequest()
                   {
                       RoleName = roleName,
                       PolicyArn = policy.PolicyArn
                   });
```

```
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteInstanceProfileà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour supprimer un profil d'instance

La commande `delete-instance-profile` suivante supprime un profil d'instance nommé `ExampleInstanceProfile`.

```
aws iam delete-instance-profile \
    --instance-profile-name ExampleInstanceProfile
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Utilisation de profils d'instance dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteInstanceProfileà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const client = new IAMClient({});
await client.send(
  new DeleteInstanceProfileCommand({
    InstanceProfileName: NAMES.instanceProfileName,
  }),
);
```

- Pour plus de détails sur l'API, reportez-vous DeleteInstanceProfileà la section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime le profil d'instance EC2 nommé **MyAppInstanceProfile**. La première commande détache tous les rôles du profil d'instance, puis la seconde commande supprime le profil d'instance.

```
(Get-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile).Roles |
 Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyAppInstanceProfile
Remove-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile
```

- Pour plus de détails sur l'API, reportez-vous DeleteInstanceProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime le profil d'instance EC2 nommé **MyAppInstanceProfile**. La première commande détache tous les rôles du profil d'instance, puis la seconde commande supprime le profil d'instance.

```
(Get-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile).Roles |
 Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyAppInstanceProfile
Remove-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile
```

- Pour plus de détails sur l'API, reportez-vous DeleteInstanceProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple supprime le rôle du profil d'instance, détache toutes les politiques associées au rôle et supprime toutes les ressources.

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
```

```
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def delete_instance_profile(self, profile_name: str, role_name: str) -> None:
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
```

```
self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            log.error(
                f"Couldn't delete instance profile {profile_name} or detach "
                f"policies and delete role {role_name}: {err}"
            )
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )
```

- Pour plus de détails sur l'API, consultez DeleteInstanceProfilele AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`DeleteLoginProfile`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `DeleteLoginProfile`.

CLI

AWS CLI

Pour supprimer le mot de passe d'un utilisateur IAM

La commande `delete-login-profile` suivante supprime le mot de passe de l'utilisateur IAM nommé Bob.

```
aws iam delete-login-profile \
    --user-name Bob
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir DeleteLoginProfilela section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime le profil de connexion de l'utilisateur IAM nommé **Bob**. Cela empêche l'utilisateur de se connecter à la AWS console. Cela n'empêche pas l'utilisateur d'exécuter des appels de AWS CLI ou d'API AWS à l'aide de clés d'accès qui peuvent toujours être associées au compte utilisateur. PowerShell

```
Remove-IAMLoginProfile -UserName Bob
```

- Pour plus de détails sur l'API, reportez-vous DeleteLoginProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime le profil de connexion de l'utilisateur IAM nommé **Bob**. Cela empêche l'utilisateur de se connecter à la AWS console. Cela n'empêche pas l'utilisateur d'exécuter des appels de AWS CLI ou d'API AWS à l'aide de clés d'accès qui peuvent toujours être associées au compte utilisateur. PowerShell

```
Remove-IAMLoginProfile -UserName Bob
```

- Pour plus de détails sur l'API, reportez-vous DeleteLoginProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`DeleteOpenIdConnectProvider`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `DeleteOpenIdConnectProvider`.

CLI

AWS CLI

Pour supprimer un fournisseur d'identité IAM OpenID Connect

Cet exemple supprime le fournisseur OIDC IAM qui se connecte au fournisseur `example.oidcprovider.com`.

```
aws iam delete-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Création de fournisseurs d'identité OpenID Connect (OIDC) du Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteOpenIdConnectProviderà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime le fournisseur IAM OIDC qui se connecte au fournisseur **`example.oidcprovider.com`**. Assurez-vous de mettre à jour ou de supprimer tous les rôles

qui font référence à ce fournisseur dans l'élément **`Principal`** de la politique d'approbation du rôle.

```
Remove-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Pour plus de détails sur l'API, reportez-vous DeleteOpenIdConnectProviderà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime le fournisseur IAM OIDC qui se connecte au fournisseur **`example.oidcprovider.com`**. Assurez-vous de mettre à jour ou de supprimer tous les rôles qui font référence à ce fournisseur dans l'élément **`Principal`** de la politique d'approbation du rôle.

```
Remove-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Pour plus de détails sur l'API, reportez-vous DeleteOpenIdConnectProviderà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`DeletePolicy`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `DeletePolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Principes de base
- Créer des utilisateurs en lecture seule et en lecture-écriture
- Création d'un espace de travail de surveillance géré
- Débuter avec le streaming géré
- Commencer à orchestrer les flux de travail

- Gérer les politiques
- Configuration de la gestion des systèmes

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des AWS SDK pour .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an AWS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
```

```
}

# Retrieve the calling parameters.
while getopts "n:h" option; do
  case "${option}" in
    n) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy arn with the -n parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Policy arn:  $policy_arn"
iecho ""

response=$(aws iam delete-policy \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
  return 1
fi

iecho "delete-policy response:$response"
iecho

return 0
```

```
}
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::deletePolicy(const Aws::String &policyArn,
                               const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::DeletePolicyRequest request;
    request.SetPolicyArn(policyArn);

    auto outcome = iam.DeletePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting policy with arn " << policyArn << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted policy with arn " << policyArn
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour supprimer une politique IAM

Cet exemple supprime la politique dont l'ARN est `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam delete-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)
```

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(ctx context.Context, policyArn string)
 error {
 _, err := wrapper.IamClient.DeletePolicy(ctx, &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des AWS SDK pour Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.DeletePolicyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeletePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <policyARN>\s

                Where:
                    policyARN - A policy ARN value to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String policyARN = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMPolicy(iam, policyARN);
        iam.close();
    }

    public static void deleteIAMPolicy(IamClient iam, String policyARN) {
        try {
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(policyARN)
                    .build();

            iam.deletePolicy(request);
            System.out.println("Successfully deleted the policy");
```

```
        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Supprimez la politique.

```
import { DeletePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 */
export const deletePolicy = (policyArn) => {
  const command = new DeletePolicyCommand({ PolicyArn: policyArn });
  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun deleteIAMPolicy(policyARNVal: String?) {
    val request =
        DeletePolicyRequest {
            policyArn = policyARNVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deletePolicy(request)
        println("Successfully deleted $policyARNVal")
    }
}
```

- Pour plus de détails sur l'API, consultez DeletePolicyla section AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime la politique dont l'ARN est **`arn:aws:iam::123456789012:policy/MySamplePolicy`**. Avant de supprimer la politique, vous devez commencer par supprimer toutes les versions, à l'exception de la politique par défaut, en exécutant **`Remove-IAMPolicyVersion`**. Vous devez également détacher la politique de tous les utilisateurs, groupes ou rôles IAM.

```
Remove-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Exemple 2 : cet exemple supprime une politique en supprimant d'abord toutes les versions de politique autres que celles par défaut, en la détachant de toutes les entités IAM auxquelles

elle est attachée, puis en supprimant la politique elle-même. La première ligne extrait l'objet de politique. La deuxième ligne extrait toutes les versions de la politique qui ne sont pas marquées comme étant la version par défaut dans une collection et supprime ensuite chaque politique de la collection. La troisième ligne extrait tous les utilisateurs, groupes et rôles IAM auxquels la politique est attachée. Les lignes quatre à six détachent la politique de chaque entité attachée. La dernière ligne utilise cette commande pour supprimer la politique gérée ainsi que la version par défaut restante. L'exemple inclut le paramètre booléen **`-Force`** sur toute ligne qui en a besoin pour supprimer les invites de confirmation.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
$attached = Get-IAMEntitiesForPolicy -PolicyArn $pol.Arn
$attached.PolicyGroups | Unregister-IAMGroupPolicy -PolicyArn $pol.arn
$attached.PolicyRoles | Unregister-IAMRolePolicy -PolicyArn $pol.arn
$attached.PolicyUsers | Unregister-IAMUserPolicy -PolicyArn $pol.arn
Remove-IAMPolicy $pol.Arn -Force
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime la politique dont l'ARN est **`arn:aws:iam::123456789012:policy/MySamplePolicy`**. Avant de supprimer la politique, vous devez commencer par supprimer toutes les versions, à l'exception de la politique par défaut, en exécutant **`Remove-IAMPolicyVersion`**. Vous devez également détacher la politique de tous les utilisateurs, groupes ou rôles IAM.

```
Remove-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Exemple 2 : cet exemple supprime une politique en supprimant d'abord toutes les versions de politique autres que celles par défaut, en la détachant de toutes les entités IAM auxquelles elle est attachée, puis en supprimant la politique elle-même. La première ligne extrait l'objet de politique. La deuxième ligne extrait toutes les versions de la politique qui ne sont pas marquées comme étant la version par défaut dans une collection et supprime ensuite chaque politique de la collection. La troisième ligne extrait tous les utilisateurs, groupes et rôles IAM auxquels la politique est attachée. Les lignes quatre à six détachent la politique de chaque entité attachée. La dernière ligne utilise cette commande pour supprimer la politique gérée

ainsi que la version par défaut restante. L'exemple inclut le paramètre booléen **`-Force`** sur toute ligne qui en a besoin pour supprimer les invites de confirmation.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
$attached = Get-IAMEntitiesForPolicy -PolicyArn $pol.Arn
$attached.PolicyGroups | Unregister-IAMGroupPolicy -PolicyArn $pol.arn
$attached.PolicyRoles | Unregister-IAMRolePolicy -PolicyArn $pol.arn
$attached.PolicyUsers | Unregister-IAMUserPolicy -PolicyArn $pol.arn
Remove-IAMPolicy $pol.Arn -Force
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

- Pour plus de détails sur l'API, consultez DeletePolicyle AWS manuel de référence de l'API SDK for Python (Boto3).

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn delete_policy(client: &iamClient, policy: Policy) -> Result<(),
 iamError> {
    client
        .delete_policy()
        .policy_arn(policy.arn.unwrap())
        .send()
        .await?;
    Ok(())
}
```

- Pour plus de détails sur l'API, voir DeletePolicyla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        lo_iam->deletepolicy( iv_policyarn = iv_policy_arn ).
```

```
        MESSAGE 'Policy deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamdeleteconflictex.
        MESSAGE 'Policy cannot be deleted due to attachments.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func deletePolicy(policy: IAMClientTypes.Policy) async throws {
        let input = DeletePolicyInput(
            policyArn: policy.arn
        )
        do {
            _ = try await iamClient.deletePolicy(input: input)
        } catch {
            print("ERROR: deletePolicy:", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`DeletePolicyVersion`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `DeletePolicyVersion`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Gérer les politiques
- Restaurer une version de stratégie

CLI

AWS CLI

Pour supprimer une version d'une politique gérée

Cet exemple supprime la version identifiée `v2` de la politique dont l'ARN est `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam delete-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir DeletePolicyVersionla section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime la version identifiée comme **v2** de la politique dont l'ARN est **arn:aws:iam::123456789012:policy/MySamplePolicy**.

```
Remove-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MySamplePolicy -VersionID v2
```

Exemple 2 : cet exemple supprime une politique en supprimant d'abord toutes les versions de politique autres que celles par défaut, puis en supprimant la politique elle-même. La première ligne extrait l'objet de politique. La deuxième ligne extrait toutes les versions de politique qui ne sont pas signalées comme étant par défaut dans une collection, puis utilise cette commande pour supprimer chaque politique de la collection. La dernière ligne supprime la politique elle-même ainsi que la version par défaut restante. Notez que pour supprimer correctement une politique gérée, vous devez également la détacher de tous les utilisateurs, groupes ou rôles à l'aide des commandes **Unregister-IAMUserPolicy**, **Unregister-IAMGroupPolicy** et **Unregister-IAMRolePolicy**. Consultez l'exemple pour l'applet de commande **Remove-IAMPolicy**.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
Remove-IAMPolicy -PolicyArn $pol.Arn -force
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyVersionà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime la version identifiée comme **v2** de la politique dont l'ARN est **arn:aws:iam::123456789012:policy/MySamplePolicy**.

```
Remove-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MySamplePolicy -VersionID v2
```

Exemple 2 : cet exemple supprime une politique en supprimant d'abord toutes les versions de politique autres que celles par défaut, puis en supprimant la politique elle-même. La première ligne extrait l'objet de politique. La deuxième ligne extrait toutes les versions de politique qui ne

sont pas signalées comme étant par défaut dans une collection, puis utilise cette commande pour supprimer chaque politique de la collection. La dernière ligne supprime la politique elle-même ainsi que la version par défaut restante. Notez que pour supprimer correctement une politique gérée, vous devez également la détacher de tous les utilisateurs, groupes ou rôles à l'aide des commandes **`Unregister-IAMUserPolicy`**, **`Unregister-IAMGroupPolicy`** et **`Unregister-IAMRolePolicy`**. Consultez l'exemple pour l'applet de commande **`Remove-IAMPolicy`**.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
Remove-IAMPolicy -PolicyArn $pol.Arn -force
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyVersionà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        lo_iam->deletepolicyversion(
          iv_policyarn = iv_policy_arn
          iv_versionid = iv_version_id ).
        MESSAGE 'Policy version deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy or version does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamdeleteconflictex.
        MESSAGE 'Cannot delete default policy version.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, consultez DeletePolicyVersionla section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`DeleteRole`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `DeleteRole`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Principes de base
- Configuration de la connectivité des services de conteneurs
- Création d'une API REST avec intégration de proxy de fonctions
- Création d'un espace de travail de surveillance géré
- Création d'un tableau de bord de surveillance avec le nom de la fonction comme variable
- Création de votre première fonction sans serveur
- Premiers pas avec les entrepôts de données sans serveur
- Commencer à gérer la configuration
- Débuter avec la protection des appareils liés à l'Internet des objets
- Commencer à utiliser les magasins de fonctionnalités d'apprentissage automatique
- Commencer à utiliser les clusters Kubernetes gérés
- Débuter avec le streaming géré
- Commencer à utiliser des clusters d'entrepôts de données provisionnés
- Commencer à orchestrer les flux de travail
- Gérer les rôles
- Transférer les secrets codés en dur vers un stockage secret sécurisé
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts
- Configuration de la gestion des systèmes

- Utilisation de variables de propriété dans les tableaux de bord de surveillance pour surveiller plusieurs fonctions sans serveur

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
{ RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section Référence des AWS SDK pour .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an AWS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
```

```
}

# Retrieve the calling parameters.
while getopts "n:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

echo "role_name:$role_name"
if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Role name:  $role_name"
iecho ""

response=$(aws iam delete-role \
  --role-name "$role_name")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-role operation failed.\n$response"
  return 1
fi

iecho "delete-role response:$response"
iecho
```

```
  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section Référence des AWS CLI commandes.

CLI

AWS CLI

Pour supprimer un rôle IAM

La commande `delete-role` suivante supprime le rôle nommé `Test-Role`.

```
aws iam delete-role \
    --role-name Test-Role
```

Cette commande ne produit aucune sortie.

Pour pouvoir supprimer un rôle, vous devez le supprimer de tout profil d'instance (`remove-role-from-instance-profile`), détacher toutes les politiques gérées (`detach-role-policy`) et supprimer toutes les politiques intégrées attachées au rôle (`delete-role-policy`).

Pour plus d'informations, consultez Création de rôles IAM et Utilisation de profils d'instance dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(ctx context.Context, roleName string) error
 {
 _, err := wrapper.IamClient.DeleteRole(ctx, &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section Référence des AWS SDK pour Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Supprimez le rôle.

```
import { DeleteRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const deleteRole = (roleName) => {
  const command = new DeleteRoleCommand({ RoleName: roleName });
  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime le rôle nommé **`MyNewRole`** du compte IAM actuel. Avant de pouvoir supprimer le rôle, vous devez d'abord utiliser la commande **`Unregister-IAMRolePolicy`** pour détacher les politiques gérées. Les politiques en ligne sont supprimées avec le rôle.

```
Remove-IAMRole -RoleName MyNewRole
```

Exemple 2 : cet exemple détache toutes les politiques gérées du rôle nommé **MyNewRole**, puis supprime le rôle. La première ligne extrait toutes les politiques gérées associées au rôle en tant que collection, puis détache du rôle chaque politique de la collection. La deuxième ligne supprime le rôle lui-même. Les politiques en ligne sont supprimées en même temps que le rôle.

```
Get-IAMAttachedRolePolicyList -RoleName MyNewRole | Unregister-IAMRolePolicy -
RoleName MyNewRole
Remove-IAMRole -RoleName MyNewRole
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime le rôle nommé **MyNewRole** du compte IAM actuel. Avant de pouvoir supprimer le rôle, vous devez d'abord utiliser la commande **Unregister-IAMRolePolicy** pour détacher les politiques gérées. Les politiques en ligne sont supprimées avec le rôle.

```
Remove-IAMRole -RoleName MyNewRole
```

Exemple 2 : cet exemple détache toutes les politiques gérées du rôle nommé **MyNewRole**, puis supprime le rôle. La première ligne extrait toutes les politiques gérées associées au rôle en tant que collection, puis détache du rôle chaque politique de la collection. La deuxième ligne supprime le rôle lui-même. Les politiques en ligne sont supprimées en même temps que le rôle.

```
Get-IAMAttachedRolePolicyList -RoleName MyNewRole | Unregister-IAMRolePolicy -
RoleName MyNewRole
Remove-IAMRole -RoleName MyNewRole
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def delete_role(role_name):
    """
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

- Pour plus de détails sur l'API, consultez DeleteRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
  # Deletes a role and its attached policies.
```

```
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    # Detach and delete attached policies
    @iam_client.list_attached_role_policies(role_name: role_name).each do |
response|
      response.attached_policies.each do |policy|
        @iam_client.detach_role_policy({
                                         role_name: role_name,
                                         policy_arn: policy.policy_arn
                                       })
        # Check if the policy is a customer managed policy (not AWS managed)
        unless policy.policy_arn.include?('aws:policy/')
          @iam_client.delete_policy({ policy_arn: policy.policy_arn })
          @logger.info("Deleted customer managed policy #{policy.policy_name}.")
        end
      end
    end

    # Delete the role
    @iam_client.delete_role({ role_name: role_name })
    @logger.info("Deleted role #{role_name}.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't detach policies and delete role #{role_name}. Here's
 why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn delete_role(client: &iamClient, role: &Role) -> Result<(), iamError>
 {
    let role = role.clone();
    while client
        .delete_role()
        .role_name(role.role_name())
        .send()
        .await
        .is_err()
    {
        sleep(Duration::from_secs(2)).await;
    }
    Ok(())
}
```

- Pour plus de détails sur l'API, voir DeleteRolela section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        lo_iam->deleterole( iv_rolename = iv_role_name ).
        MESSAGE 'Role deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamdeleteconflictex.
        MESSAGE 'Role cannot be deleted due to attached resources.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func deleteRole(role: IAMClientTypes.Role) async throws {
        let input = DeleteRoleInput(
            roleName: role.roleName
        )
        do {
            _ = try await iamClient.deleteRole(input: input)
        } catch {
            print("ERROR: deleteRole:", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`DeleteRolePermissionsBoundary`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `DeleteRolePermissionsBoundary`.

CLI

AWS CLI

Pour supprimer une limite des autorisations d'un rôle IAM

L'exemple `delete-role-permissions-boundary` suivant supprime la limite des autorisations pour le rôle IAM spécifié. Pour appliquer une limite des autorisations à un rôle, utilisez la commande `put-role-permissions-boundary`.

```
aws iam delete-role-permissions-boundary \
    --role-name lambda-application-role
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteRolePermissionsBoundaryà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple montre comment supprimer la limite des autorisations attachée à un rôle IAM.

```
Remove-IAMRolePermissionsBoundary -RoleName MyRoleName
```

- Pour plus de détails sur l'API, reportez-vous DeleteRolePermissionsBoundaryà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple montre comment supprimer la limite des autorisations attachée à un rôle IAM.

```
Remove-IAMRolePermissionsBoundary -RoleName MyRoleName
```

- Pour plus de détails sur l'API, reportez-vous DeleteRolePermissionsBoundaryà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`DeleteRolePolicy`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `DeleteRolePolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Premiers pas avec les entrepôts de données sans serveur
- Commencer à gérer la configuration
- Débuter avec la protection des appareils liés à l'Internet des objets
- Commencer à utiliser des clusters d'entrepôts de données provisionnés
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
```

```
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteRolePolicyà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour supprimer une politique d'un rôle IAM

La commande `delete-role-policy` suivante supprime la politique nommée `ExamplePolicy` du rôle nommé `Test-Role`.

```
aws iam delete-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Modification d'un rôle dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteRolePolicyà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { DeleteRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 * @param {string} policyName
 */
export const deleteRolePolicy = (roleName, policyName) => {
  const command = new DeleteRolePolicyCommand({
    RoleName: roleName,
    PolicyName: policyName,
  });
  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous DeleteRolePolicyà la section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime la politique en ligne **`S3AccessPolicy`** intégrée au rôle IAM **`S3BackupRole`**.

```
Remove-IAMRolePolicy -PolicyName S3AccessPolicy -RoleName S3BackupRole
```

- Pour plus de détails sur l'API, reportez-vous DeleteRolePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime la politique en ligne **S3AccessPolicy** intégrée au rôle IAM **S3BackupRole**.

```
Remove-IAMRolePolicy -PolicyName S3AccessPolicy -RoleName S3BackupRole
```

- Pour plus de détails sur l'API, reportez-vous DeleteRolePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisez **DeleteSAMLProvider avec un** AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `DeleteSAMLProvider`.

CLI

AWS CLI

Pour supprimer un fournisseur SAML

Cet exemple supprime le fournisseur IAM SAML 2.0 dont l'ARN est `arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider`.

```
aws iam delete-saml-provider \
--saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Création de fournisseurs d'identité SAML IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez DeleteSAMLProvider dans la Référence des commandes de l'AWS CLI .

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { DeleteSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} providerArn
 * @returns
 */
export const deleteSAMLProvider = async (providerArn) => {
  const command = new DeleteSAMLProviderCommand({
    SAMLProviderArn: providerArn,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Pour les détails de l'API, consultez DeleteSAMLProvider dans la Référence de l'API AWS SDK pour JavaScript .

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime le fournisseur IAM SAML 2.0 dont l'ARN est **`arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider`**.

```
Remove-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFSProvider
```

- Pour plus de détails sur l'API, consultez DeleteSAMLProvider dans la Référence des applets de commande pour les Outils AWS pour PowerShell (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime le fournisseur IAM SAML 2.0 dont l'ARN est **`arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider`**.

```
Remove-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFSProvider
```

- Pour plus de détails sur l'API, consultez DeleteSAMLProvider dans la Référence des applets de commande pour les Outils AWS pour PowerShell (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`DeleteServerCertificate`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `DeleteServerCertificate`.

C++

Kit de développement logiciel (SDK) for C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::deleteServerCertificate(const Aws::String &certificateName,
                                          const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::DeleteServerCertificateRequest request;
    request.SetServerCertificateName(certificateName);
```

```
    const auto outcome = iam.DeleteServerCertificate(request);
    bool result = true;
    if (!outcome.IsSuccess()) {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error deleting server certificate " << certificateName
 <<
                      ": " << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << certificateName
                      << "' not found." << std::endl;
        }
    }
    else {
        std::cout << "Successfully deleted server certificate " <<
 certificateName
                  << std::endl;
    }

    return result;
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteServerCertificateà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour supprimer un certificat de serveur de votre AWS compte

La `delete-server-certificate` commande suivante supprime le certificat de serveur spécifié de votre AWS compte.

```
aws iam delete-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

Cette commande ne produit aucune sortie.

Pour répertorier les certificats de serveur disponibles dans votre AWS compte, utilisez la `list-server-certificates` commande.

Pour plus d'informations, consultez Gestion des certificats de serveur dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteServerCertificateà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Supprimez un certificat de serveur.

```
import { DeleteServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 */
export const deleteServerCertificate = (certName) => {
  const command = new DeleteServerCertificateCommand({
    ServerCertificateName: certName,
  });

  return client.send(command);
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous DeleteServerCertificateà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteServerCertificate(
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
);
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous DeleteServerCertificateà la section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime le certificat de serveur nommé **`MyServerCert`**.

```
Remove-IAMServerCertificate -ServerCertificateName MyServerCert
```

- Pour plus de détails sur l'API, reportez-vous DeleteServerCertificateà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime le certificat de serveur nommé **MyServerCert**.

```
Remove-IAMServerCertificate -ServerCertificateName MyServerCert
```

- Pour plus de détails sur l'API, reportez-vous DeleteServerCertificateà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez, mettez à jour et supprimez des certificats de serveur.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'ServerCertificateManager'
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key
                                          })
```

```
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info('No server certificates found.')
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
#{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
'#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
```

```
    false
  end
end
```

- Pour plus de détails sur l'API, reportez-vous DeleteServerCertificateà la section Référence des AWS SDK pour Ruby API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`DeleteServiceLinkedRole`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `DeleteServiceLinkedRole`.

CLI

AWS CLI

Pour supprimer un rôle lié à un service

L'exemple `delete-service-linked-role` suivant supprime le rôle lié à un service spécifié dont vous n'avez plus besoin. La suppression s'effectue de manière asynchrone. Vous pouvez vérifier le statut de la suppression et si elle est terminée à l'aide de la commande `get-service-linked-role-deletion-status`.

```
aws iam delete-service-linked-role \
    --role-name AWSServiceRoleForLexBots
```

Sortie :

```
{
    "DeletionTaskId": "task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE"
}
```

Pour plus d'informations, consultez Utilisation des rôles liés à un service dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteServiceLinkedRoleà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(ctx context.Context, roleName
 string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(ctx,
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
```

```
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}
```

- Pour plus de détails sur l'API, voir DeleteServiceLinkedRolela section Référence des AWS SDK pour Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { DeleteServiceLinkedRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const deleteServiceLinkedRole = (roleName) => {
  const command = new DeleteServiceLinkedRoleCommand({ RoleName: roleName });
  return client.send(command);
};
```

- Pour plus de détails sur l'API, voir DeleteServiceLinkedRolela section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime le rôle lié au service. Notez que si le service utilise toujours ce rôle, l'exécution de cette commande se soldera par un échec.

```
Remove-IAMServiceLinkedRole -RoleName
 AWSServiceRoleForAutoScaling_RoleNameEndsWithThis
```

- Pour plus de détails sur l'API, reportez-vous DeleteServiceLinkedRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime le rôle lié au service. Notez que si le service utilise toujours ce rôle, l'exécution de cette commande se soldera par un échec.

```
Remove-IAMServiceLinkedRole -RoleName
 AWSServiceRoleForAutoScaling_RoleNameEndsWithThis
```

- Pour plus de détails sur l'API, reportez-vous DeleteServiceLinkedRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
  # Deletes a service-linked role.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_service_linked_role(role_name)
    response = @iam_client.delete_service_linked_role(role_name: role_name)
```

```
    task_id = response.deletion_task_id
    check_deletion_status(role_name, task_id)
  rescue Aws::Errors::ServiceError => e
    handle_deletion_error(e, role_name)
  end

  private

  # Checks the deletion status of a service-linked role
  #
  # @param role_name [String] The name of the role being deleted
  # @param task_id [String] The task ID for the deletion process
  def check_deletion_status(role_name, task_id)
    loop do
      response = @iam_client.get_service_linked_role_deletion_status(
        deletion_task_id: task_id
      )
      status = response.status
      @logger.info("Deletion of #{role_name} #{status}.")
      break if %w[SUCCEEDED FAILED].include?(status)

      sleep(3)
    end
  end

  # Handles deletion error
  #
  # @param e [Aws::Errors::ServiceError] The error encountered during deletion
  # @param role_name [String] The name of the role attempted to delete
  def handle_deletion_error(e, role_name)
    return if e.code == 'NoSuchEntity'

    @logger.error("Couldn't delete #{role_name}. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- Pour plus de détails sur l'API, voir DeleteServiceLinkedRolela section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn delete_service_linked_role(
    client: &iamClient,
    role_name: &str,
) -> Result<(), iamError> {
    client
        .delete_service_linked_role()
        .role_name(role_name)
        .send()
        .await?;

    Ok(())
}
```

- Pour plus de détails sur l'API, voir DeleteServiceLinkedRolela section de référence de l'API AWS SDK for Rust.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`DeleteSigningCertificate`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `DeleteSigningCertificate`.

CLI

AWS CLI

Pour supprimer un certificat de signature d'un utilisateur IAM

La commande `delete-signing-certificate` suivante supprime le certificat de signature spécifié pour l'utilisateur IAM nommé Bob.

```
aws iam delete-signing-certificate \
    --user-name Bob \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE
```

Cette commande ne produit aucune sortie.

Pour obtenir l'ID d'un certificat de signature, utilisez la commande `list-signing-certificates`.

Pour plus d'informations, consultez Gestion des certificats de signature dans le Guide de l'utilisateur Amazon EC2.

- Pour plus de détails sur l'API, reportez-vous DeleteSigningCertificateà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime le certificat de signature portant l'ID **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** de l'utilisateur IAM nommé. **Bob**

```
Remove-IAMSigningCertificate -UserName Bob -CertificateId
 Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
```

- Pour plus de détails sur l'API, reportez-vous DeleteSigningCertificateà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime le certificat de signature portant l'ID **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** de l'utilisateur IAM nommé. **Bob**

```
Remove-IAMSigningCertificate -UserName Bob -CertificateId
 Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
```

- Pour plus de détails sur l'API, reportez-vous DeleteSigningCertificateà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`DeleteUser`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `DeleteUser`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Principes de base
- Créer des utilisateurs en lecture seule et en lecture-écriture

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS SDK pour .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
```

```
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""
```

```
  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteUserRequest request;
```

```
    request.SetUserName(userName);
    auto outcome = iam.DeleteUser(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting IAM user " << userName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;;
    }
    else {
        std::cout << "Successfully deleted IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour supprimer un utilisateur IAM

La commande `delete-user` suivante supprime l'utilisateur IAM nommé Bob du compte actuel.

```
aws iam delete-user \
    --user-name Bob
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Suppression d'un utilisateur IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(ctx context.Context, userName string) error
 {
 _, err := wrapper.IamClient.DeleteUser(ctx, &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS SDK pour Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteUserRequest;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <userName>\s

                Where:
                    userName - The name of the user to delete.\s
                """;
```

```
        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMUser(iam, userName);
        System.out.println("Done");
        iam.close();
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("Successfully deleted IAM user " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Supprimez l'utilisateur.

```
import { DeleteUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} name
 */
export const deleteUser = (name) => {
  const command = new DeleteUserCommand({ UserName: name });
  return client.send(command);
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
```

```
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    console.log("User " + process.argv[2] + " does not exist.");
  } else {
    iam.deleteUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
      } else {
        console.log("Success", data);
      }
    });
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun deleteIAMUser(userNameVal: String) {
    val request =
```

```
        DeleteUserRequest {
            userName = userNameVal
        }

    // To delete a user, ensure that the user's access keys are deleted first.
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteUser(request)
        println("Successfully deleted user $userNameVal")
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime l'utilisateur IAM nommé **Bob**.

```
Remove-IAMUser -UserName Bob
```

Exemple 2 : cet exemple supprime l'utilisateur IAM nommé **Theresa**, ainsi que tous les éléments qui doivent être supprimés au préalable.

```
$name = "Theresa"

# find any groups and remove user from them
$groups = Get-IAMGroupForUser -UserName $name
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName $name -Force }

# find any inline policies and delete them
$inlinepols = Get-IAMUserPolicies -UserName $name
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 $name -Force}

# find any managed polices and detach them
$managedpols = Get-IAMAttachedUserPolicies -UserName $name
foreach ($pol in $managedpols) { Unregister-IAMUserPolicy -PolicyArn
 $pol.PolicyArn -UserName $name }
```

```
# find any signing certificates and delete them
$certs = Get-IAMSigningCertificate -UserName $name
foreach ($cert in $certs) { Remove-IAMSigningCertificate -CertificateId
 $cert.CertificateId -UserName $name -Force }

# find any access keys and delete them
$keys = Get-IAMAccessKey -UserName $name
foreach ($key in $keys) { Remove-IAMAccessKey -AccessKeyId $key.AccessKeyId -
UserName $name -Force }

# delete the user's login profile, if one exists - note: need to use try/catch to
 suppress not found error
try { $prof = Get-IAMLoginProfile -UserName $name -ea 0 } catch { out-null }
if ($prof) { Remove-IAMLoginProfile -UserName $name -Force }

# find any MFA device, detach it, and if virtual, delete it.
$mfa = Get-IAMMFADevice -UserName $name
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}

# finally, remove the user
Remove-IAMUser -UserName $name -Force
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime l'utilisateur IAM nommé **Bob**.

```
Remove-IAMUser -UserName Bob
```

Exemple 2 : cet exemple supprime l'utilisateur IAM nommé **Theresa**, ainsi que tous les éléments qui doivent être supprimés au préalable.

```
$name = "Theresa"

# find any groups and remove user from them
$groups = Get-IAMGroupForUser -UserName $name
```

```
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName $name -Force }

# find any inline policies and delete them
$inlinepols = Get-IAMUserPolicies -UserName $name
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 $name -Force}

# find any managed polices and detach them
$managedpols = Get-IAMAttachedUserPolicies -UserName $name
foreach ($pol in $managedpols) { Unregister-IAMUserPolicy -PolicyArn
 $pol.PolicyArn -UserName $name }

# find any signing certificates and delete them
$certs = Get-IAMSigningCertificate -UserName $name
foreach ($cert in $certs) { Remove-IAMSigningCertificate -CertificateId
 $cert.CertificateId -UserName $name -Force }

# find any access keys and delete them
$keys = Get-IAMAccessKey -UserName $name
foreach ($key in $keys) { Remove-IAMAccessKey -AccessKeyId $key.AccessKeyId -
UserName $name -Force }

# delete the user's login profile, if one exists - note: need to use try/catch to
 suppress not found error
try { $prof = Get-IAMLoginProfile -UserName $name -ea 0 } catch { out-null }
if ($prof) { Remove-IAMLoginProfile -UserName $name -Force }

# find any MFA device, detach it, and if virtual, delete it.
$mfa = Get-IAMMFADevice -UserName $name
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}

# finally, remove the user
Remove-IAMUser -UserName $name -Force
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def delete_user(user_name):
    """
    Deletes a user. Before a user can be deleted, all associated resources,
    such as access keys and policies, must be deleted or detached.

    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
        raise
```

- Pour plus de détails sur l'API, consultez DeleteUserle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
  # Deletes a user and their associated resources
  #
  # @param user_name [String] The name of the user to delete
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn delete_user(client: &iamClient, user: &User) -> Result<(),
 SdkError<DeleteUserError>> {
    let user = user.clone();
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;

    let response: Result<(), SdkError<DeleteUserError>> = loop {
        match client
            .delete_user()
```

```
            .user_name(user.user_name())
            .send()
            .await
        {
            Ok(_) => {
                break Ok(());
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    response
}
```

- Pour plus de détails sur l'API, voir DeleteUserla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        lo_iam->deleteuser( iv_username = iv_user_name ).
        MESSAGE 'User deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamdeleteconflictex.
        MESSAGE 'User cannot be deleted due to attached resources.' TYPE 'E'.
```

```
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func deleteUser(user: IAMClientTypes.User) async throws {
        let input = DeleteUserInput(
            userName: user.userName
        )
        do {
            _ = try await iamClient.deleteUser(input: input)
        } catch {
            print("ERROR: deleteUser:", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`DeleteUserPermissionsBoundary`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `DeleteUserPermissionsBoundary`.

CLI

AWS CLI

Pour supprimer une limite des autorisations d'un utilisateur IAM

L'exemple `delete-user-permissions-boundary` suivant supprime la limite des autorisations attachée à l'utilisateur IAM nommé `intern`. Pour appliquer une limite des autorisations à un utilisateur, utilisez la commande `put-user-permissions-boundary`.

```
aws iam delete-user-permissions-boundary \
    --user-name intern
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteUserPermissionsBoundaryà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple montre comment supprimer la limite des autorisations attachée à un utilisateur IAM.

```
Remove-IAMUserPermissionsBoundary -UserName joe
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserPermissionsBoundaryà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple montre comment supprimer la limite des autorisations attachée à un utilisateur IAM.

```
Remove-IAMUserPermissionsBoundary -UserName joe
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserPermissionsBoundaryà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`DeleteUserPolicy`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `DeleteUserPolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });
```

```
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserPolicyà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour supprimer une politique d'un utilisateur IAM

La commande `delete-user-policy` suivante supprime la politique spécifiée de l'utilisateur IAM nommé Bob.

```
aws iam delete-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

Cette commande ne produit aucune sortie.

Pour obtenir une liste des politiques d'un utilisateur IAM, utilisez la commande `list-user-policies`.

Pour plus d'informations, consultez la section Création d'un utilisateur IAM dans votre AWS compte dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteUserPolicyà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(ctx context.Context, userName string,
 policyName string) error {
 _, err := wrapper.IamClient.DeleteUserPolicy(ctx, &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
```

```
 return err
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserPolicyà la section Référence des AWS SDK pour Go API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime la politique en ligne nommée **AccessToEC2Policy** intégrée dans le nom de l'utilisateur IAM **Bob**.

```
Remove-IAMUserPolicy -PolicyName AccessToEC2Policy -UserName Bob
```

Exemple 2 : cet exemple recherche toutes les politiques en ligne qui sont intégrées dans l'utilisateur IAM nommé **Theresa**, puis les supprime.

```
$inlinepols = Get-IAMUserPolicies -UserName Theresa
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 Theresa -Force}
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime la politique en ligne nommée **AccessToEC2Policy** intégrée dans le nom de l'utilisateur IAM **Bob**.

```
Remove-IAMUserPolicy -PolicyName AccessToEC2Policy -UserName Bob
```

Exemple 2 : cet exemple recherche toutes les politiques en ligne qui sont intégrées dans l'utilisateur IAM nommé **Theresa**, puis les supprime.

```
$inlinepols = Get-IAMUserPolicies -UserName Theresa
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 Theresa -Force}
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Deletes a user and their associated resources
 #
 # @param user_name [String] The name of the user to delete
 def delete_user(user_name)
   user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
   user.each do |key|
     @iam_client.delete_access_key({ access_key_id: key.access_key_id,
user_name: user_name })
     @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
   end

   @iam_client.delete_user(user_name: user_name)
   @logger.info("Deleted user '#{user_name}'.")
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error deleting user '#{user_name}': #{e.message}")
 end
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserPolicyà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn delete_user_policy(
    client: &iamClient,
    user: &User,
    policy_name: &str,
) -> Result<(), SdkError<DeleteUserPolicyError>> {
    client
        .delete_user_policy()
        .user_name(user.user_name())
        .policy_name(policy_name)
        .send()
        .await?;

    Ok(())
}
```

- Pour plus de détails sur l'API, voir DeleteUserPolicyla section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
```

```
import AWSS3


   func deleteUserPolicy(user: IAMClientTypes.User, policyName: String) async
 throws {
        let input = DeleteUserPolicyInput(
            policyName: policyName,
            userName: user.userName
        )
        do {
            _ = try await iamClient.deleteUserPolicy(input: input)
        } catch {
            print("ERROR: deleteUserPolicy:", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserPolicyà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`DeleteVirtualMfaDevice`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `DeleteVirtualMfaDevice`.

CLI

AWS CLI

Pour retirer un dispositif MFA virtuel

La commande `delete-virtual-mfa-device` suivante supprime le dispositif MFA spécifié du compte actuel.

```
aws iam delete-virtual-mfa-device \
    --serial-number arn:aws:iam::123456789012:mfa/MFATest
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Désactivation des dispositifs MFA dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteVirtualMfaDeviceà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime le dispositif MFA virtuel IAM dont l'ARN est **`arn:aws:iam::123456789012:mfa/bob`**.

```
Remove-IAMVirtualMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/bob
```

Exemple 2 : cet exemple vérifie si l'utilisateur IAM Theresa s'est vu attribuer un dispositif MFA. S'il en trouve un, le dispositif est désactivé pour l'utilisateur IAM. Si le dispositif est virtuel, il est également supprimé.

```
$mfa = Get-IAMMFADevice -UserName Theresa
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteVirtualMfaDeviceà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime le dispositif MFA virtuel IAM dont l'ARN est **`arn:aws:iam::123456789012:mfa/bob`**.

```
Remove-IAMVirtualMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/bob
```

Exemple 2 : cet exemple vérifie si l'utilisateur IAM Theresa s'est vu attribuer un dispositif MFA. S'il en trouve un, le dispositif est désactivé pour l'utilisateur IAM. Si le dispositif est virtuel, il est également supprimé.

```
$mfa = Get-IAMMFADevice -UserName Theresa
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteVirtualMfaDeviceà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`DetachGroupPolicy`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `DetachGroupPolicy`.

CLI

AWS CLI

Pour détacher une politique d'un groupe

Cet exemple supprime la politique gérée, dont l'ARN est `arn:aws:iam::123456789012:policy/TesterAccessPolicy`, du rôle nommé `Testers`.

```
aws iam detach-group-policy \
    --group-name Testers \
    --policy-arn arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Gestion des groupes IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DetachGroupPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple détache la politique de groupe géré dont l'ARN est **`arn:aws:iam::123456789012:policy/TesterAccessPolicy`** du groupe nommé **`Testers`**.

```
Unregister-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Exemple 2 : cet exemple trouve toutes les politiques gérées associées au groupe nommé **`Testers`** et les détache du groupe.

```
Get-IAMAttachedGroupPolicies -GroupName Testers | Unregister-IAMGroupPolicy -
Groupname Testers
```

- Pour plus de détails sur l'API, reportez-vous DetachGroupPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple détache la politique de groupe géré dont l'ARN est **`arn:aws:iam::123456789012:policy/TesterAccessPolicy`** du groupe nommé **`Testers`**.

```
Unregister-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Exemple 2 : cet exemple trouve toutes les politiques gérées associées au groupe nommé **`Testers`** et les détache du groupe.

```
Get-IAMAttachedGroupPolicies -GroupName Testers | Unregister-IAMGroupPolicy -
Groupname Testers
```

- Pour plus de détails sur l'API, reportez-vous DetachGroupPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`DetachRolePolicy`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `DetachRolePolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Principes de base
- Configuration de la connectivité des services de conteneurs
- Création d'une API REST avec intégration de proxy de fonctions
- Création d'un espace de travail de surveillance géré
- Création d'un tableau de bord de surveillance avec le nom de la fonction comme variable
- Création de votre première fonction sans serveur
- Commencer à gérer la configuration
- Débuter avec la protection des appareils liés à l'Internet des objets
- Commencer à utiliser les magasins de fonctionnalités d'apprentissage automatique
- Commencer à utiliser les clusters Kubernetes gérés
- Débuter avec le streaming géré
- Commencer à orchestrer les flux de travail
- Gérer les rôles
- Transférer les secrets codés en dur vers un stockage secret sécurisé
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts
- Configuration de la gestion des systèmes
- Utilisation de variables de propriété dans les tableaux de bord de surveillance pour surveiller plusieurs fonctions sans serveur

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous DetachRolePolicyà la section Référence des AWS SDK pour .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
```

```
  echo "  -p policy_ARN -- The IAM policy document ARN."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy ARN with the -p parameter."
  usage
  return 1
fi

response=$(aws iam detach-role-policy \
  --role-name "$role_name" \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
  return 1
```

```
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous DetachRolePolicyà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DetachRolePolicyRequest detachRequest;
    detachRequest.SetRoleName(roleName);
    detachRequest.SetPolicyArn(policyArn);

    auto detachOutcome = iam.DetachRolePolicy(detachRequest);
    if (!detachOutcome.IsSuccess()) {
        std::cerr << "Failed to detach policy " << policyArn << " from role "
                  << roleName << ": " << detachOutcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully detached policy " << policyArn << " from role
 "
                  << roleName << std::endl;
    }

    return detachOutcome.IsSuccess();
```

- Pour plus de détails sur l'API, reportez-vous DetachRolePolicyà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour détacher une politique d'un rôle

Cet exemple supprime la politique gérée avec l'ARN `arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy` du rôle nommé `FedTesterRole`.

```
aws iam detach-role-policy \
    --role-name FedTesterRole \
    --policy-arn arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Modification d'un rôle dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DetachRolePolicyà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
```

```
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(ctx context.Context, roleName string,
 policyArn string) error {
 _, err := wrapper.IamClient.DetachRolePolicy(ctx, &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}
```

- Pour plus de détails sur l'API, reportez-vous DetachRolePolicyà la section Référence des AWS SDK pour Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.DetachRolePolicyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DetachRolePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }
```

```
        String roleName = args[0];
        String policyArn = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        detachPolicy(iam, roleName, policyArn);
        System.out.println("Done");
        iam.close();
    }

    public static void detachPolicy(IamClient iam, String roleName, String
 policyArn) {
        try {
            DetachRolePolicyRequest request = DetachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.detachRolePolicy(request);
            System.out.println("Successfully detached policy " + policyArn +
                    " from role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DetachRolePolicyà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Détachez la politique.

```
import { DetachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 * @param {string} roleName
 */
export const detachRolePolicy = (policyArn, roleName) => {
  const command = new DetachRolePolicyCommand({
    PolicyArn: policyArn,
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous DetachRolePolicyà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        var params = {
          PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
          RoleName: process.argv[2],
        };
        iam.detachRolePolicy(params, function (err, data) {
          if (err) {
            console.log("Unable to detach policy from role", err);
          } else {
            console.log("Policy detached from role successfully");
            process.exit();
          }
        });
      }
    });
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous DetachRolePolicyà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun detachPolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        DetachRolePolicyRequest {
            roleName = roleNameVal
            policyArn = policyArnVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.detachRolePolicy(request)
        println("Successfully detached policy $policyArnVal from role
 $roleNameVal")
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DetachRolePolicyà la section AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple détache la politique de groupe géré dont l'ARN est **`arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy`** du rôle nommé **`FedTesterRole`**.

```
Unregister-IAMRolePolicy -RoleName FedTesterRole -PolicyArn
 arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Exemple 2 : cet exemple recherche toutes les politiques gérées associées au rôle nommé **`FedTesterRole`** et les détache du rôle.

```
Get-IAMAttachedRolePolicyList -RoleName FedTesterRole | Unregister-IAMRolePolicy
 -Rolename FedTesterRole
```

- Pour plus de détails sur l'API, reportez-vous DetachRolePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple détache la politique de groupe géré dont l'ARN est **`arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy`** du rôle nommé **`FedTesterRole`**.

```
Unregister-IAMRolePolicy -RoleName FedTesterRole -PolicyArn
 arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Exemple 2 : cet exemple recherche toutes les politiques gérées associées au rôle nommé **`FedTesterRole`** et les détache du rôle.

```
Get-IAMAttachedRolePolicyList -RoleName FedTesterRole | Unregister-IAMRolePolicy
 -Rolename FedTesterRole
```

- Pour plus de détails sur l'API, reportez-vous DetachRolePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Détachez une politique d'un rôle à l'aide de l'objet de politique Boto3.

```
def detach_from_role(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Policy(policy_arn).detach_role(RoleName=role_name)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

Détachez une politique d'un rôle à l'aide de l'objet de rôle Boto3.

```
def detach_policy(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

- Pour plus de détails sur l'API, consultez DetachRolePolicyle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, attache et détache les politiques de rôle.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end
```

```
 # Fetches an IAM policy by its ARN
 # @param policy_arn [String] the ARN of the IAM policy to retrieve
 # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
 def get_policy(policy_arn)
   response = @iam_client.get_policy(policy_arn: policy_arn)
   policy = response.policy
   @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
   policy
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
   raise
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
   raise
 end

 # Attaches a policy to a role
 #
 # @param role_name [String] The name of the role
 # @param policy_arn [String] The policy ARN
 # @return [Boolean] true if successful, false otherwise
 def attach_policy_to_role(role_name, policy_arn)
   @iam_client.attach_role_policy(
     role_name: role_name,
     policy_arn: policy_arn
   )
   true
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error attaching policy to role: #{e.message}")
   false
 end

 # Lists policy ARNs attached to a role
 #
 # @param role_name [String] The name of the role
 # @return [Array<String>] List of policy ARNs
 def list_attached_policy_arns(role_name)
   response = @iam_client.list_attached_role_policies(role_name: role_name)
   response.attached_policies.map(&:policy_arn)
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error listing policies attached to role: #{e.message}")
```

```
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, reportez-vous DetachRolePolicyà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn detach_role_policy(
    client: &iamClient,
    role_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .detach_role_policy()
```

```
        .role_name(role_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- Pour plus de détails sur l'API, voir DetachRolePolicyla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        lo_iam->detachrolepolicy(
          iv_rolename = iv_role_name
          iv_policyarn = iv_policy_arn ).
        MESSAGE 'Policy detached from role successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role or policy does not exist.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous DetachRolePolicyà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func detachRolePolicy(policy: IAMClientTypes.Policy, role:
 IAMClientTypes.Role) async throws {
        let input = DetachRolePolicyInput(
            policyArn: policy.arn,
            roleName: role.roleName
        )

        do {
            _ = try await iamClient.detachRolePolicy(input: input)
        } catch {
            print("ERROR: detachRolePolicy:", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous DetachRolePolicyà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`DetachUserPolicy`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `DetachUserPolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Créer des utilisateurs en lecture seule et en lecture-écriture

CLI

AWS CLI

Pour détacher une politique d'un utilisateur

Cet exemple supprime la politique gérée avec l'ARN `arn:aws:iam::123456789012:policy/TesterPolicy` de l'utilisateur Bob.

```
aws iam detach-user-policy \
    --user-name Bob \
    --policy-arn arn:aws:iam::123456789012:policy/TesterPolicy
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Modification des autorisations pour un utilisateur IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir DetachUserPolicyla section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple détache la politique gérée dont l'ARN est **`arn:aws:iam::123456789012:policy/TesterPolicy`** de l'utilisateur IAM nommé **`Bob`**.

```
Unregister-IAMUserPolicy -UserName Bob -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Exemple 2 : cet exemple trouve toutes les politiques gérées associées à l'utilisateur IAM nommé **`Theresa`** et détache ces politiques de l'utilisateur.

```
Get-IAMAttachedUserPolicyList -UserName Theresa | Unregister-IAMUserPolicy -
Username Theresa
```

- Pour plus de détails sur l'API, reportez-vous DetachUserPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple détache la politique gérée dont l'ARN est **`arn:aws:iam::123456789012:policy/TesterPolicy`** de l'utilisateur IAM nommé **`Bob`**.

```
Unregister-IAMUserPolicy -UserName Bob -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Exemple 2 : cet exemple trouve toutes les politiques gérées associées à l'utilisateur IAM nommé **`Theresa`** et détache ces politiques de l'utilisateur.

```
Get-IAMAttachedUserPolicyList -UserName Theresa | Unregister-IAMUserPolicy -
Username Theresa
```

- Pour plus de détails sur l'API, reportez-vous DetachUserPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
```

```
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from user %s.", policy_arn, user_name
        )
        raise
```

- Pour plus de détails sur l'API, consultez DetachUserPolicyle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Detaches a policy from a user
 #
 # @param user_name [String] The name of the user
 # @param policy_arn [String] The ARN of the policy to detach
 # @return [Boolean] true if the policy was successfully detached, false
otherwise
 def detach_user_policy(user_name, policy_arn)
   @iam_client.detach_user_policy(
     user_name: user_name,
     policy_arn: policy_arn
   )
   @logger.info("Policy '#{policy_arn}' detached from user '#{user_name}'
successfully.")
   true
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error('Error detaching policy: Policy or user does not exist.')
```

```
    false
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from user '#{user_name}':
 #{e.message}")
    false
  end
```

- Pour plus de détails sur l'API, voir DetachUserPolicyla section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn detach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .detach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- Pour plus de détails sur l'API, voir DetachUserPolicyla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    lo_iam->detachuserpolicy(
      iv_username = iv_user_name
      iv_policyarn = iv_policy_arn ).
    MESSAGE 'Policy detached from user successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'User or policy does not exist.' TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, consultez DetachUserPolicyla section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`EnableMfaDevice`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `EnableMfaDevice`.

CLI

AWS CLI

Pour activer un dispositif MFA

Après avoir utilisé la `create-virtual-mfa-device` commande pour créer un dispositif MFA virtuel, vous pouvez attribuer le dispositif MFA à un utilisateur. L'exemple `enable-mfa-device` suivant attribue le dispositif MFA, dont le numéro de série est

`arn:aws:iam::210987654321:mfa/BobsMFADevice`, à l'utilisateur Bob. La commande synchronise également le dispositif AWS en incluant les deux premiers codes en séquence à partir du dispositif MFA virtuel.

```
aws iam enable-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 789012
```

Cette commande ne produit aucune sortie.

Pour de plus d'informations, consultez Activation d'un dispositif virtuel d'authentification multifactorielle (MFA) dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous EnableMfaDeviceà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cette commande active le dispositif MFA matériel dont le numéro de série est **987654321098** et l'associe à l'utilisateur **Bob**. Elle inclut les deux premiers codes en séquence provenant du dispositif.

```
Enable-IAMMFADevice -UserName "Bob" -SerialNumber "987654321098" -
AuthenticationCode1 "12345678" -AuthenticationCode2 "87654321"
```

Exemple 2 : cet exemple crée et active un dispositif MFA virtuel. La première commande crée le dispositif virtuel et renvoie la représentation de l'objet du dispositif dans la variable **$MFADevice**. Vous pouvez utiliser les propriétés **.Base32StringSeed** ou **QRCodePng** pour configurer l'application logicielle de l'utilisateur. La commande finale attribue le dispositif à l'utilisateur **David**, en identifiant le dispositif par son numéro de série. La commande synchronise également le dispositif AWS en incluant les deux premiers codes en séquence à partir du dispositif MFA virtuel.

```
$MFADevice = New-IAMVirtualMFADevice -VirtualMFADeviceName "MyMFADevice"
# see example for New-IAMVirtualMFADevice to see how to configure the software
 program with PNG or base32 seed code
```

```
Enable-IAMMFADevice -UserName "David" -SerialNumber -SerialNumber
 $MFADevice.SerialNumber -AuthenticationCode1 "24681357" -AuthenticationCode2
 "13572468"
```

- Pour plus de détails sur l'API, reportez-vous EnableMfaDeviceà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cette commande active le dispositif MFA matériel dont le numéro de série est **987654321098** et l'associe à l'utilisateur **Bob**. Elle inclut les deux premiers codes en séquence provenant du dispositif.

```
Enable-IAMMFADevice -UserName "Bob" -SerialNumber "987654321098" -
AuthenticationCode1 "12345678" -AuthenticationCode2 "87654321"
```

Exemple 2 : cet exemple crée et active un dispositif MFA virtuel. La première commande crée le dispositif virtuel et renvoie la représentation de l'objet du dispositif dans la variable **$MFADevice**. Vous pouvez utiliser les propriétés **.Base32StringSeed** ou **QRCodePng** pour configurer l'application logicielle de l'utilisateur. La commande finale attribue le dispositif à l'utilisateur **David**, en identifiant le dispositif par son numéro de série. La commande synchronise également le dispositif AWS en incluant les deux premiers codes en séquence à partir du dispositif MFA virtuel.

```
$MFADevice = New-IAMVirtualMFADevice -VirtualMFADeviceName "MyMFADevice"
# see example for New-IAMVirtualMFADevice to see how to configure the software
 program with PNG or base32 seed code
Enable-IAMMFADevice -UserName "David" -SerialNumber -SerialNumber
 $MFADevice.SerialNumber -AuthenticationCode1 "24681357" -AuthenticationCode2
 "13572468"
```

- Pour plus de détails sur l'API, reportez-vous EnableMfaDeviceà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **GenerateCredentialReport**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `GenerateCredentialReport`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer votre compte

CLI

AWS CLI

Pour générer un rapport sur les informations d'identification

L'exemple suivant tente de générer un rapport d'identification pour le AWS compte.

```
aws iam generate-credential-report
```

Sortie :

```
{
    "State":  "STARTED",
    "Description": "No report exists. Starting a new report generation task"
}
```

Pour plus d'informations, consultez la section Obtenir des rapports d'identification pour votre AWS compte dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GenerateCredentialReportà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple demande la génération d'un nouveau rapport, qui peut être effectué toutes les quatre heures. Si le dernier rapport est encore récent, le champ État est libellé comme suit : **`COMPLETE`**. Utilisez **`Get-IAMCredentialReport`** pour afficher le rapport complété.

```
Request-IAMCredentialReport
```

Sortie :

```
Description                                                    State
-----------                                                    -----
No report exists. Starting a new report generation task        STARTED
```

- Pour plus de détails sur l'API, reportez-vous GenerateCredentialReportà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple demande la génération d'un nouveau rapport, qui peut être effectué toutes les quatre heures. Si le dernier rapport est encore récent, le champ État est libellé comme suit : **`COMPLETE`**. Utilisez **`Get-IAMCredentialReport`** pour afficher le rapport complété.

```
Request-IAMCredentialReport
```

Sortie :

```
Description                                                    State
-----------                                                    -----
No report exists. Starting a new report generation task        STARTED
```

- Pour plus de détails sur l'API, reportez-vous GenerateCredentialReportà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def generate_credential_report():
    """
```

```
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
    to get the latest report. A new report can be generated a minimum of four
hours
    after the last one was generated.
    """
    try:
        response = iam.meta.client.generate_credential_report()
        logger.info(
            "Generating credentials report for your account. " "Current state is
%s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
account.")
        raise
    else:
        return response
```

- Pour plus de détails sur l'API, consultez GenerateCredentialReportle AWS manuel de référence de l'API SDK for Python (Boto3).

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->generatecredentialreport( ).
        MESSAGE 'Credential report generation started.' TYPE 'I'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Report generation limit exceeded.' TYPE 'E'.
```

```
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when generating credential report.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous GenerateCredentialReportà la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`GenerateServiceLastAccessedDetails`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `GenerateServiceLastAccessedDetails`.

CLI

AWS CLI

Exemple 1 : pour générer un rapport d'accès aux services pour une politique personnalisée

L'exemple `generate-service-last-accessed-details` suivant lance une tâche en arrière-plan pour générer un rapport répertoriant les services auxquels accèdent les utilisateurs IAM et les autres entités avec une politique personnalisée nommée `intern-boundary`. Vous pouvez afficher le rapport une fois qu'il a été créé en exécutant la commande `get-service-last-accessed-details`.

```
aws iam generate-service-last-accessed-details \
    --arn arn:aws:iam::123456789012:policy/intern-boundary
```

Sortie :

```
{
    "JobId": "2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc"
}
```

Exemple 2 : pour générer un rapport d'accès aux services pour la AdministratorAccess politique AWS gérée

L'`generate-service-last-accessed-details`exemple suivant lance une tâche en arrière-plan pour générer un rapport répertoriant les services auxquels accèdent les utilisateurs IAM et les autres entités avec la `AdministratorAccess` politique AWS gérée. Vous pouvez afficher le rapport une fois qu'il a été créé en exécutant la commande `get-service-last-accessed-details`.

```
aws iam generate-service-last-accessed-details \
    --arn arn:aws:iam::aws:policy/AdministratorAccess
```

Sortie :

```
{
    "JobId": "78b6c2ba-d09e-6xmp-7039-ecde30b26916"
}
```

Pour plus d'informations, voir Affiner les autorisations lors de AWS l'utilisation des dernières informations consultées dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GenerateServiceLastAccessedDetailsà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple est une applet de commande équivalente à l' GenerateServiceLastAccessedDetails API. Cela fournit un identifiant de tâche qui peut être utilisé dans Get-IAMServiceLastAccessedDetail et Get-IAMServiceLastAccessedDetailWithEntity

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

- Pour plus de détails sur l'API, reportez-vous GenerateServiceLastAccessedDetailsà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple est une applet de commande équivalente à l' GenerateServiceLastAccessedDetails API. Cela fournit un identifiant de

tâche qui peut être utilisé dans Get-IAMServiceLastAccessedDetail et Get-IAMServiceLastAccessedDetailWithEntity

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

- Pour plus de détails sur l'API, reportez-vous GenerateServiceLastAccessedDetailsà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`GetAccessKeyLastUsed`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `GetAccessKeyLastUsed`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer des clés d'accès

C++

SDK pour C++

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::accessKeyLastUsed(const Aws::String &secretKeyID,
                                    const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetAccessKeyLastUsedRequest request;

    request.SetAccessKeyId(secretKeyID);
```

```
    Aws::IAM::Model::GetAccessKeyLastUsedOutcome outcome =
 iam.GetAccessKeyLastUsed(
            request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error querying last used time for access key " <<
                  secretKeyID << ":" << outcome.GetError().GetMessage() <<
 std::endl;
    }
    else {
        Aws::String lastUsedTimeString =
                outcome.GetResult()
                        .GetAccessKeyLastUsed()
                        .GetLastUsedDate()
                        .ToGmtString(Aws::Utils::DateFormat::ISO_8601);
        std::cout << "Access key " << secretKeyID << " last used at time " <<
                  lastUsedTimeString << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, voir GetAccessKeyLastUsedla section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour récupérer les informations relatives au moment où la clé d'accès spécifiée a été utilisée pour la dernière fois

L'exemple suivant récupère les informations relatives au moment où la clé d'accès ABCDEXAMPLE a été utilisée pour la dernière fois.

```
aws iam get-access-key-last-used \
    --access-key-id ABCDEXAMPLE
```

Sortie :

```
{
```

```
    "UserName":  "Bob",
    "AccessKeyLastUsed": {
        "Region": "us-east-1",
        "ServiceName": "iam",
        "LastUsedDate": "2015-06-16T22:45:00Z"
    }
}
```

Pour plus d'informations, consultez Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetAccessKeyLastUsedà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Obtenez la clé d'accès.

```
import { GetAccessKeyLastUsedCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} accessKeyId
 */
export const getAccessKeyLastUsed = async (accessKeyId) => {
  const command = new GetAccessKeyLastUsedCommand({
    AccessKeyId: accessKeyId,
  });

  const response = await client.send(command);

  if (response.AccessKeyLastUsed?.LastUsedDate) {
```

```
    console.log(`
    ${accessKeyId} was last used by ${response.UserName} via
    the ${response.AccessKeyLastUsed.ServiceName} service on
    ${response.AccessKeyLastUsed.LastUsedDate.toISOString()}
    `);
  }

  return response;
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, voir GetAccessKeyLastUsedla section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.getAccessKeyLastUsed(
  { AccessKeyId: "ACCESS_KEY_ID" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data.AccessKeyLastUsed);
    }
  }
);
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, voir GetAccessKeyLastUsedla section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : renvoie le nom d'utilisateur propriétaire et les informations relatives à la dernière utilisation de la clé d'accès fournie.

```
Get-IAMAccessKeyLastUsed -AccessKeyId ABCDEXAMPLE
```

- Pour plus de détails sur l'API, reportez-vous GetAccessKeyLastUsedà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : renvoie le nom d'utilisateur propriétaire et les informations relatives à la dernière utilisation de la clé d'accès fournie.

```
Get-IAMAccessKeyLastUsed -AccessKeyId ABCDEXAMPLE
```

- Pour plus de détails sur l'API, reportez-vous GetAccessKeyLastUsedà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def get_last_use(key_id):
    """
    Gets information about when and how a key was last used.
```

```
    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
    try:
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
            "Key %s was last used by %s on %s to access %s.",
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
    except ClientError:
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
        return response
```

- Pour plus de détails sur l'API, consultez GetAccessKeyLastUsedle AWS manuel de référence de l'API SDK for Python (Boto3).

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->getaccesskeylastused(
          iv_accesskeyid = iv_access_key_id ).
        MESSAGE 'Retrieved access key last used information.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
```

```
        MESSAGE 'Access key does not exist.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, consultez GetAccessKeyLastUsedla section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`GetAccountAuthorizationDetails`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `GetAccountAuthorizationDetails`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer votre compte

CLI

AWS CLI

Pour répertorier les utilisateurs, les groupes, les rôles et les politiques IAM d'un AWS compte

La `get-account-authorization-details` commande suivante renvoie des informations sur tous les utilisateurs, groupes, rôles et politiques IAM du AWS compte.

```
aws iam get-account-authorization-details
```

Sortie :

```
{
    "RoleDetailList": [
        {
            "AssumeRolePolicyDocument": {
                "Version":"2012-10-17",
                "Statement": [
                    {
```

```
                    "Sid": "",
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "ec2.amazonaws.com"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        },
        "RoleId": "AROA1234567890EXAMPLE",
        "CreateDate": "2014-07-30T17:09:20Z",
        "InstanceProfileList": [
            {
                "InstanceProfileId": "AIPA1234567890EXAMPLE",
                "Roles": [
                    {
                        "AssumeRolePolicyDocument": {
                            "Version":"2012-10-17",
                            "Statement": [
                                {
                                    "Sid": "",
                                    "Effect": "Allow",
                                    "Principal": {
                                        "Service": "ec2.amazonaws.com"
                                    },
                                    "Action": "sts:AssumeRole"
                                }
                            ]
                        },
                        "RoleId": "AROA1234567890EXAMPLE",
                        "CreateDate": "2014-07-30T17:09:20Z",
                        "RoleName": "EC2role",
                        "Path": "/",
                        "Arn": "arn:aws:iam::123456789012:role/EC2role"
                    }
                ],
                "CreateDate": "2014-07-30T17:09:20Z",
                "InstanceProfileName": "EC2role",
                "Path": "/",
                "Arn": "arn:aws:iam::123456789012:instance-profile/EC2role"
            }
        ],
        "RoleName": "EC2role",
        "Path": "/",
```

```
            "AttachedManagedPolicies": [
                {
                    "PolicyName": "AmazonS3FullAccess",
                    "PolicyArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
                },
                {
                    "PolicyName": "AmazonDynamoDBFullAccess",
                    "PolicyArn": "arn:aws:iam::aws:policy/
AmazonDynamoDBFullAccess"
                }
            ],
            "RoleLastUsed": {
                "Region": "us-west-2",
                "LastUsedDate": "2019-11-13T17:30:00Z"
            },
            "RolePolicyList": [],
            "Arn": "arn:aws:iam::123456789012:role/EC2role"
        }
    ],
    "GroupDetailList": [
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "AdministratorAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
            },
            "GroupName": "Admins",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "CreateDate": "2013-10-14T18:32:24Z",
            "GroupPolicyList": []
        },
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "PowerUserAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/PowerUserAccess"
            },
            "GroupName": "Dev",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Dev",
            "CreateDate": "2013-10-14T18:33:55Z",
            "GroupPolicyList": []
        },
```

```
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": [],
            "GroupName": "Finance",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Finance",
            "CreateDate": "2013-10-14T18:57:48Z",
            "GroupPolicyList": [
                {
                    "PolicyName": "policygen-201310141157",
                    "PolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Action": "aws-portal:*",
                                "Sid": "Stmt1381777017000",
                                "Resource": "*",
                                "Effect": "Allow"
                            }
                        ]
                    }
                }
            ]
        }
    ],
    "UserDetailList": [
        {
            "UserName": "Alice",
            "GroupList": [
                "Admins"
            ],
            "CreateDate": "2013-10-14T18:32:24Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Alice"
        },
        {
            "UserName": "Bob",
            "GroupList": [
                "Admins"
            ],
            "CreateDate": "2013-10-14T18:32:25Z",
```

```
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [
                {
                    "PolicyName": "DenyBillingAndIAMPolicy",
                    "PolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": {
                            "Effect": "Deny",
                            "Action": [
                                "aws-portal:*",
                                "iam:*"
                            ],
                            "Resource": "*"
                        }
                    }
                }
            ],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Bob"
        },
        {
            "UserName": "Charlie",
            "GroupList": [
                "Dev"
            ],
            "CreateDate": "2013-10-14T18:33:56Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Charlie"
        }
    ],
    "Policies": [
        {
            "PolicyName": "create-update-delete-set-managed-policies",
            "CreateDate": "2015-02-06T19:58:34Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
```

```
                    "CreateDate": "2015-02-06T19:58:34Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version":"2012-10-17",
                        "Statement": {
                            "Effect": "Allow",
                            "Action": [
                                "iam:CreatePolicy",
                                "iam:CreatePolicyVersion",
                                "iam:DeletePolicy",
                                "iam:DeletePolicyVersion",
                                "iam:GetPolicy",
                                "iam:GetPolicyVersion",
                                "iam:ListPolicies",
                                "iam:ListPolicyVersions",
                                "iam:SetDefaultPolicyVersion"
                            ],
                            "Resource": "*"
                        }
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/create-update-delete-set-
managed-policies",
            "UpdateDate": "2015-02-06T19:58:34Z"
        },
        {
            "PolicyName": "S3-read-only-specific-bucket",
            "CreateDate": "2015-01-21T21:39:41Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2015-01-21T21:39:41Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
```

```
                                "Action": [
                                    "s3:Get*",
                                    "s3:List*"
                                ],
                                "Resource": [
                                    "arn:aws:s3:::amzn-s3-demo-bucket",
                                    "arn:aws:s3:::amzn-s3-demo-bucket/*"
                                ]
                            }
                        ]
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/S3-read-only-specific-
bucket",
            "UpdateDate": "2015-01-21T23:39:41Z"
        },
        {
            "PolicyName": "AmazonEC2FullAccess",
            "CreateDate": "2015-02-06T18:40:15Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2014-10-30T20:59:46Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Action": "ec2:*",
                                "Effect": "Allow",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "elasticloadbalancing:*",
                                "Resource": "*"
                            },
                            {
```

```
                                "Effect": "Allow",
                                "Action": "cloudwatch:*",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "autoscaling:*",
                                "Resource": "*"
                            }
                        ]
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::aws:policy/AmazonEC2FullAccess",
            "UpdateDate": "2015-02-06T18:40:15Z"
        }
    ],
    "Marker": "EXAMPLEkakv9BCuUNFDtxWSyfzetYwEx2ADc8dnzfvERF5S6YMvXKx41t6gCl/
eeaCX3Jo94/bKqezEAg8TEVS99EKFLxm3jtbpl25FDWEXAMPLE",
    "IsTruncated": true
}
```

Pour plus d'informations, consultez Consignes pour les audits de sécuritéAWS dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetAccountAuthorizationDetailsà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple obtient les détails d'autorisation concernant les identités du AWS compte et affiche la liste des éléments de l'objet renvoyé, y compris les utilisateurs, les groupes et les rôles. Par exemple, la propriété **`UserDetailList`** affiche des informations sur les utilisateurs. Des informations similaires sont disponibles dans les propriétés **`RoleDetailList`** et **`GroupDetailList`**.

```
$Details=Get-IAMAccountAuthorizationDetail
$Details
```

Sortie :

```
GroupDetailList : {Administrators, Developers, Testers, Backup}
IsTruncated     : False
Marker          :
RoleDetailList  : {TestRole1, AdminRole, TesterRole, clirole...}
UserDetailList  : {Administrator, Bob, BackupToS3, }
```

```
$Details.UserDetailList
```

Sortie :

```
Arn             : arn:aws:iam::123456789012:user/Administrator
CreateDate      : 10/16/2014 9:03:09 AM
GroupList       : {Administrators}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE1
UserName        : Administrator
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/Bob
CreateDate      : 4/6/2015 12:54:42 PM
GroupList       : {Developers}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE2
UserName        : bab
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/BackupToS3
CreateDate      : 1/27/2015 10:15:08 AM
GroupList       : {Backup}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE3
UserName        : BackupToS3
UserPolicyList : {BackupServicePermissionsToS3Buckets}
```

- Pour plus de détails sur l'API, reportez-vous GetAccountAuthorizationDetailsà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple obtient les détails d'autorisation concernant les identités du AWS compte et affiche la liste des éléments de l'objet renvoyé, y compris les utilisateurs, les

groupes et les rôles. Par exemple, la propriété **`UserDetailList`** affiche des informations sur les utilisateurs. Des informations similaires sont disponibles dans les propriétés **`RoleDetailList`** et **`GroupDetailList`**.

```
$Details=Get-IAMAccountAuthorizationDetail
$Details
```

Sortie :

```
GroupDetailList : {Administrators, Developers, Testers, Backup}
IsTruncated     : False
Marker          :
RoleDetailList  : {TestRole1, AdminRole, TesterRole, clirole...}
UserDetailList  : {Administrator, Bob, BackupToS3, }
```

```
$Details.UserDetailList
```

Sortie :

```
Arn             : arn:aws:iam::123456789012:user/Administrator
CreateDate      : 10/16/2014 9:03:09 AM
GroupList       : {Administrators}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE1
UserName        : Administrator
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/Bob
CreateDate      : 4/6/2015 12:54:42 PM
GroupList       : {Developers}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE2
UserName        : bab
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/BackupToS3
CreateDate      : 1/27/2015 10:15:08 AM
GroupList       : {Backup}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE3
UserName        : BackupToS3
```

```
UserPolicyList : {BackupServicePermissionsToS3Buckets}
```

- Pour plus de détails sur l'API, reportez-vous GetAccountAuthorizationDetailsà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def get_authorization_details(response_filter):
    """
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
 such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

- Pour plus de détails sur l'API, consultez GetAccountAuthorizationDetailsle AWS manuel de référence de l'API SDK for Python (Boto3).

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   TRY.
       oo_result = lo_iam->getaccountauthdetails( ).
       MESSAGE 'Retrieved account authorization details.' TYPE 'I'.
     CATCH /aws1/cx_iamservicefailureex.
       MESSAGE 'Service failure when getting account authorization details.'
TYPE 'E'.
   ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous GetAccountAuthorizationDetailsà la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`GetAccountPasswordPolicy`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `GetAccountPasswordPolicy`.

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }
```

- Pour plus de détails sur l'API, voir GetAccountPasswordPolicyla section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour afficher la politique de mot de passe du compte actuel

La commande `get-account-password-policy` suivante affiche les détails de la politique de mot de passe du compte actuel.

```
aws iam get-account-password-policy
```

Sortie :

```
{
    "PasswordPolicy": {
        "AllowUsersToChangePassword": false,
        "RequireLowercaseCharacters": false,
        "RequireUppercaseCharacters": false,
        "MinimumPasswordLength": 8,
        "RequireNumbers": true,
        "RequireSymbols": true
    }
}
```

Si aucune politique de mot de passe n'est définie pour le compte, la commande renvoie une erreur `NoSuchEntity`.

Pour plus d'informations, consultez Définition d'une politique de mot de passe du compte pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir GetAccountPasswordPolicyla section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
```

```
func (wrapper AccountWrapper) GetAccountPasswordPolicy(ctx context.Context)
 (*types.PasswordPolicy, error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(ctx,
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}
```

- Pour plus de détails sur l'API, voir GetAccountPasswordPolicyla section Référence des AWS SDK pour Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Obtenez la politique de mot de passe du compte.

```
import {
  GetAccountPasswordPolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const getAccountPasswordPolicy = async () => {
  const command = new GetAccountPasswordPolicyCommand({});

  const response = await client.send(command);
```

```
  console.log(response.PasswordPolicy);
  return response;
};
```

- Pour plus de détails sur l'API, voir GetAccountPasswordPolicyla section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getAccountPasswordPolicy()
    {
        return $this->iamClient->getAccountPasswordPolicy();
    }
```

- Pour plus de détails sur l'API, voir GetAccountPasswordPolicyla section Référence des AWS SDK pour PHP API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple affiche les détails de la politique de mot de passe du compte actuel. Si aucune politique de mot de passe n'est définie pour le compte, la commande renvoie une erreur **`NoSuchEntity`**.

```
Get-IAMAccountPasswordPolicy
```

Sortie :

```
AllowUsersToChangePassword : True
ExpirePasswords            : True
HardExpiry                 : False
MaxPasswordAge             : 90
MinimumPasswordLength      : 8
PasswordReusePrevention    : 20
RequireLowercaseCharacters : True
RequireNumbers             : True
RequireSymbols             : False
RequireUppercaseCharacters : True
```

- Pour plus de détails sur l'API, reportez-vous GetAccountPasswordPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple affiche les détails de la politique de mot de passe du compte actuel. Si aucune politique de mot de passe n'est définie pour le compte, la commande renvoie une erreur **`NoSuchEntity`**.

```
Get-IAMAccountPasswordPolicy
```

Sortie :

```
AllowUsersToChangePassword : True
ExpirePasswords            : True
HardExpiry                 : False
MaxPasswordAge             : 90
MinimumPasswordLength      : 8
PasswordReusePrevention    : 20
RequireLowercaseCharacters : True
RequireNumbers             : True
RequireSymbols             : False
RequireUppercaseCharacters : True
```

- Pour plus de détails sur l'API, reportez-vous GetAccountPasswordPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def print_password_policy():
    """
    Prints the password policy for the account.
    """
    try:
        pw_policy = iam.AccountPasswordPolicy()
        print("Current account password policy:")
        print(
            f"\tallow_users_to_change_password:
 {pw_policy.allow_users_to_change_password}"
        )
        print(f"\texpire_passwords: {pw_policy.expire_passwords}")
        print(f"\thard_expiry: {pw_policy.hard_expiry}")
        print(f"\tmax_password_age: {pw_policy.max_password_age}")
        print(f"\tminimum_password_length: {pw_policy.minimum_password_length}")
        print(f"\tpassword_reuse_prevention:
 {pw_policy.password_reuse_prevention}")
        print(
            f"\trequire_lowercase_characters:
 {pw_policy.require_lowercase_characters}"
        )
        print(f"\trequire_numbers: {pw_policy.require_numbers}")
        print(f"\trequire_symbols: {pw_policy.require_symbols}")
        print(
            f"\trequire_uppercase_characters:
 {pw_policy.require_uppercase_characters}"
        )
        printed = True
    except ClientError as error:
        if error.response["Error"]["Code"] == "NoSuchEntity":
            print("The account does not have a password policy set.")
        else:
```

```
            logger.exception("Couldn't get account password policy.")
            raise
    else:
        return printed
```

- Pour plus de détails sur l'API, consultez GetAccountPasswordPolicyle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# Class to manage IAM account password policies
class PasswordPolicyManager
  attr_accessor :iam_client, :logger

  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'IAMPolicyManager'
  end

  # Retrieves and logs the account password policy
  def print_account_password_policy
    response = @iam_client.get_account_password_policy
    @logger.info("The account password policy is:
 #{response.password_policy.to_h}")
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.info('The account does not have a password policy.')
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't print the account password policy. Error: #{e.code} -
 #{e.message}")
```

```
    raise
  end
end
```

- Pour plus de détails sur l'API, voir GetAccountPasswordPolicyla section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn get_account_password_policy(
    client: &iamClient,
) -> Result<GetAccountPasswordPolicyOutput,
 SdkError<GetAccountPasswordPolicyError>> {
    let response = client.get_account_password_policy().send().await?;

    Ok(response)
}
```

- Pour plus de détails sur l'API, voir GetAccountPasswordPolicyla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->getaccountpasswordpolicy( ).
        MESSAGE 'Retrieved account password policy.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'No password policy exists.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when getting password policy.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, consultez GetAccountPasswordPolicyla section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`GetAccountSummary`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `GetAccountSummary`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer votre compte

CLI

AWS CLI

Pour récupérer les informations sur l'utilisation des entités IAM et les quotas IAM dans le compte actuel

La commande `get-account-summary` suivante renvoie des informations sur l'utilisation actuelle des entités IAM et les quotas IAM actuels dans le compte.

```
aws iam get-account-summary
```

Sortie :

```
{
    "SummaryMap": {
        "UsersQuota": 5000,
        "GroupsQuota": 100,
        "InstanceProfiles": 6,
        "SigningCertificatesPerUserQuota": 2,
        "AccountAccessKeysPresent": 0,
        "RolesQuota": 250,
        "RolePolicySizeQuota": 10240,
        "AccountSigningCertificatesPresent": 0,
        "Users": 27,
        "ServerCertificatesQuota": 20,
        "ServerCertificates": 0,
        "AssumeRolePolicySizeQuota": 2048,
        "Groups": 7,
        "MFADevicesInUse": 1,
        "Roles": 3,
        "AccountMFAEnabled": 1,
        "MFADevices": 3,
        "GroupsPerUserQuota": 10,
        "GroupPolicySizeQuota": 5120,
        "InstanceProfilesQuota": 100,
        "AccessKeysPerUserQuota": 2,
        "Providers": 0,
        "UserPolicySizeQuota": 2048
    }
}
```

Pour plus d'informations sur les limites des entités, consultez les quotas IAM et AWS STS dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetAccountSummaryà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie des informations sur l'utilisation des entités IAM et sur les quotas IAM actuels dans Compte AWS.

```
Get-IAMAccountSummary
```

Sortie :

```
Key                                     Value
Users                                   7
GroupPolicySizeQuota                    5120
PolicyVersionsInUseQuota                10000
ServerCertificatesQuota                 20
AccountSigningCertificatesPresent       0
AccountAccessKeysPresent                0
Groups                                  3
UsersQuota                              5000
RolePolicySizeQuota                     10240
UserPolicySizeQuota                     2048
GroupsPerUserQuota                      10
AssumeRolePolicySizeQuota               2048
AttachedPoliciesPerGroupQuota           2
Roles                                   9
VersionsPerPolicyQuota                  5
GroupsQuota                             100
PolicySizeQuota                         5120
Policies                                5
RolesQuota                              250
ServerCertificates                      0
AttachedPoliciesPerRoleQuota            2
MFADevicesInUse                         2
PoliciesQuota                           1000
AccountMFAEnabled                       1
Providers                               2
InstanceProfilesQuota                   100
MFADevices                              4
AccessKeysPerUserQuota                  2
AttachedPoliciesPerUserQuota            2
SigningCertificatesPerUserQuota         2
PolicyVersionsInUse                     4
InstanceProfiles                        1
...
```

- Pour plus de détails sur l'API, reportez-vous GetAccountSummaryà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie des informations sur l'utilisation des entités IAM et sur les quotas IAM actuels dans Compte AWS.

```
Get-IAMAccountSummary
```

Sortie :

```
Key                                Value
Users                              7
GroupPolicySizeQuota               5120
PolicyVersionsInUseQuota           10000
ServerCertificatesQuota            20
AccountSigningCertificatesPresent  0
AccountAccessKeysPresent           0
Groups                             3
UsersQuota                         5000
RolePolicySizeQuota                10240
UserPolicySizeQuota                2048
GroupsPerUserQuota                 10
AssumeRolePolicySizeQuota          2048
AttachedPoliciesPerGroupQuota      2
Roles                              9
VersionsPerPolicyQuota             5
GroupsQuota                        100
PolicySizeQuota                    5120
Policies                           5
RolesQuota                         250
ServerCertificates                 0
AttachedPoliciesPerRoleQuota       2
MFADevicesInUse                    2
PoliciesQuota                      1000
AccountMFAEnabled                  1
Providers                          2
InstanceProfilesQuota              100
MFADevices                         4
AccessKeysPerUserQuota             2
AttachedPoliciesPerUserQuota       2
SigningCertificatesPerUserQuota    2
PolicyVersionsInUse                4
InstanceProfiles                   1
```

```
...
```

- Pour plus de détails sur l'API, reportez-vous GetAccountSummaryà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def get_summary():
    """
    Gets a summary of account usage.

    :return: The summary of account usage.
    """
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
        raise
    else:
        return summary.summary_map
```

- Pour plus de détails sur l'API, consultez GetAccountSummaryle AWS manuel de référence de l'API SDK for Python (Boto3).

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    oo_result = lo_iam->getaccountsummary( ).
    MESSAGE 'Retrieved account summary.' TYPE 'I'.
  CATCH /aws1/cx_iamservicefailureex.
    MESSAGE 'Service failure when getting account summary.' TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous GetAccountSummaryà la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`GetContextKeysForCustomPolicy`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `GetContextKeysForCustomPolicy`.

CLI

AWS CLI

Exemple 1 : pour répertorier les clés de contexte référencées par une ou plusieurs politiques JSON personnalisées fournies en paramètre sur la ligne de commande

La commande `get-context-keys-for-custom-policy` suivante analyse chaque politique fournie et répertorie les clés de contexte utilisées par ces politiques. Utilisez cette commande pour identifier les valeurs de clé de contexte que vous devez fournir pour utiliser correctement les commandes `simulate-custom-policy` et `simulate-custom-policy`

du simulateur de politiques. Vous pouvez également récupérer la liste des clés de contexte utilisées par toutes les politiques associées à un utilisateur ou à un rôle IAM à l'aide de la commande `get-context-keys-for-custom-policy`. Les valeurs des paramètres qui commencent par `file://` ordonnent à la commande de lire le fichier et d'utiliser son contenu comme valeur du paramètre au lieu du nom du fichier lui-même.

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/${aws:username}","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
```

Sortie :

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Exemple 2 : pour répertorier les clés de contexte référencées par une ou plusieurs politiques JSON personnalisées fournies en entrée de fichier

La commande `get-context-keys-for-custom-policy` suivante est identique à l'exemple précédent, sauf que les politiques sont fournies dans un fichier plutôt que sous forme de paramètre. Comme la commande attend une liste de chaînes JSON, et non une liste de structures JSON, le fichier doit être structuré comme suit, bien que vous puissiez le réduire en une seule unité.

```
[
    "Policy1",
    "Policy2"
]
```

Ainsi, un fichier contenant la politique de l'exemple précédent doit ressembler à ce qui suit. Vous devez échapper chaque guillemet double intégré dans la chaîne de politique en le faisant précéder d'une barre oblique inverse ''.

```
[ "{\"Version\": \"2012-10-17\", \"Statement\": {\"Effect\": \"Allow
\", \"Action\": \"dynamodb:*\", \"Resource\": \"arn:aws:dynamodb:us-
west-2:128716708097:table/${aws:username}\", \"Condition\": {\"DateGreaterThan\":
 {\"aws:CurrentTime\": \"2015-08-16T12:00:00Z\"}}}}" ]
```

Ce fichier peut ensuite être soumis à la commande suivante.

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list file://policyfile.json
```

Sortie :

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Pour plus d'informations, consultez la section Utilisation du simulateur de politique IAM (AWS CLI et AWS API) dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetContextKeysForCustomPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple extrait toutes les clés de contexte présentes dans le JSON de la politique fourni. Afin de fournir plusieurs politiques, vous pouvez spécifier une liste de valeurs séparées par des virgules.

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
```

```
Get-IAMContextKeysForCustomPolicy -PolicyInputList $policy1,$policy2
```

- Pour plus de détails sur l'API, reportez-vous GetContextKeysForCustomPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple extrait toutes les clés de contexte présentes dans le JSON de la politique fourni. Afin de fournir plusieurs politiques, vous pouvez spécifier une liste de valeurs séparées par des virgules.

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForCustomPolicy -PolicyInputList $policy1,$policy2
```

- Pour plus de détails sur l'API, reportez-vous GetContextKeysForCustomPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`GetContextKeysForPrincipalPolicy`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `GetContextKeysForPrincipalPolicy`.

CLI

AWS CLI

Pour répertorier les clés de contexte référencées par toutes les politiques associées à un principal IAM

La commande `get-context-keys-for-principal-policy` suivante extrait toutes les politiques associées à l'utilisateur `saanvi` et aux groupes dont elle est membre. Elle analyse ensuite chacune d'elles et répertorie les clés de contexte utilisées par ces politiques.

Utilisez cette commande pour identifier les valeurs de clé de contexte que vous devez fournir pour utiliser correctement les commandes `simulate-custom-policy` et `simulate-principal-policy`. Vous pouvez également récupérer la liste des clés de contexte utilisées par une politique JSON arbitraire à l'aide de la commande `get-context-keys-for-custom-policy`.

```
aws iam get-context-keys-for-principal-policy \
   --policy-source-arn arn:aws:iam::123456789012:user/saanvi
```

Sortie :

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Pour plus d'informations, consultez la section Utilisation du simulateur de politique IAM (AWS CLI et AWS API) dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetContextKeysForPrincipalPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple récupère toutes les clés de contexte présentes dans le json de politique fourni et les politiques attachées à l'entité IAM (user/role etc.). Pour : PolicyInputList vous pouvez fournir une liste de valeurs multiples sous forme de valeurs séparées par des virgules.

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
```

```
Get-IAMContextKeysForPrincipalPolicy -PolicyInputList $policy1,$policy2 -
PolicySourceArn arn:aws:iam::852640994763:user/TestUser
```

- Pour plus de détails sur l'API, reportez-vous GetContextKeysForPrincipalPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple récupère toutes les clés de contexte présentes dans le json de politique fourni et les politiques attachées à l'entité IAM (user/role etc.). Pour : PolicyInputList vous pouvez fournir une liste de valeurs multiples sous forme de valeurs séparées par des virgules.

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForPrincipalPolicy -PolicyInputList $policy1,$policy2 -
PolicySourceArn arn:aws:iam::852640994763:user/TestUser
```

- Pour plus de détails sur l'API, reportez-vous GetContextKeysForPrincipalPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`GetCredentialReport`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `GetCredentialReport`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer votre compte

CLI

AWS CLI

Pour obtenir un rapport sur les informations d'identification

Cet exemple ouvre le rapport renvoyé et le transmet au pipeline sous la forme d'un tableau de lignes de texte.

```
aws iam get-credential-report
```

Sortie :

```
{
    "GeneratedTime":  "2015-06-17T19:11:50Z",
    "ReportFormat": "text/csv"
}
```

Pour plus d'informations, consultez la section Obtenir des rapports d'identification pour votre AWS compte dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetCredentialReportà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple ouvre le rapport renvoyé et le transmet au pipeline sous la forme d'un tableau de lignes de texte. La première ligne est l'en-tête avec les noms de colonnes séparés par des virgules. Chaque ligne successive est la ligne de détail d'un utilisateur, chaque champ étant séparé du suivant par une virgule. Avant de pouvoir consulter le rapport, vous devez le générer à l'aide de l'applet de commande **`Request-IAMCredentialReport`**. Pour récupérer le rapport sous forme de chaîne unique, utilisez à la **`-Raw`** place de **`-AsTextArray`**. L'alias **`-SplitLines`** est également accepté pour le commutateur **`-AsTextArray`**. Pour obtenir la liste complète des colonnes dans sortie, consultez la référence de l'API du service. Notez que si vous n'utilisez pas **`-AsTextArray`** ou **`-SplitLines`**, vous devez extraire le texte de la propriété **`.Content`** à l'aide de la classe .NET **`StreamReader`**.

```
Request-IAMCredentialReport
```

Sortie :

```
Description                                                                  State
-----------                                                                  -----
No report exists. Starting a new report generation task                      STARTED
```

```
Get-IAMCredentialReport -AsTextArray
```

Sortie :

```
 user,arn,user_creation_time,password_enabled,password_last_used,password_last_changed,pa
 root_account,arn:aws:iam::123456789012:root,2014-10-15T16:31:25+00:00,not_supported,2015
A,false,N/A,false,N/A,false,N/A
Administrator,arn:aws:iam::123456789012:user/
Administrator,2014-10-16T16:03:09+00:00,true,2015-04-20T15:18:32+00:00,2014-10-16T16:06:0
A,false,true,2014-12-03T18:53:41+00:00,true,2015-03-25T20:38:14+00:00,false,N/
A,false,N/A
Bill,arn:aws:iam::123456789012:user/Bill,2015-04-15T18:27:44+00:00,false,N/A,N/
A,N/A,false,false,N/A,false,N/A,false,2015-04-20T20:00:12+00:00,false,N/A
```

- Pour plus de détails sur l'API, reportez-vous GetCredentialReportà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple ouvre le rapport renvoyé et le transmet au pipeline sous la forme d'un tableau de lignes de texte. La première ligne est l'en-tête avec les noms de colonnes séparés par des virgules. Chaque ligne successive est la ligne de détail d'un utilisateur, chaque champ étant séparé du suivant par une virgule. Avant de pouvoir consulter le rapport, vous devez le générer à l'aide de l'applet de commande **`Request-IAMCredentialReport`**. Pour récupérer le rapport sous forme de chaîne unique, utilisez à la **`-Raw`** place de **`-AsTextArray`**. L'alias **`-SplitLines`** est également accepté pour le commutateur **`-AsTextArray`**. Pour obtenir la liste complète des colonnes dans sortie, consultez la référence de l'API du service. Notez que si vous n'utilisez pas **`-AsTextArray`** ou **`-SplitLines`**, vous devez extraire le texte de la propriété **`.Content`** à l'aide de la classe .NET **`StreamReader`**.

```
Request-IAMCredentialReport
```

Sortie :

```
Description                                                     State
-----------                                                     -----
No report exists. Starting a new report generation task         STARTED
```

```
Get-IAMCredentialReport -AsTextArray
```

Sortie :

```
 user,arn,user_creation_time,password_enabled,password_last_used,password_last_changed,pa
 root_account,arn:aws:iam::123456789012:root,2014-10-15T16:31:25+00:00,not_supported,2015
A,false,N/A,false,N/A,false,N/A
Administrator,arn:aws:iam::123456789012:user/
Administrator,2014-10-16T16:03:09+00:00,true,2015-04-20T15:18:32+00:00,2014-10-16T16:06:0
A,false,true,2014-12-03T18:53:41+00:00,true,2015-03-25T20:38:14+00:00,false,N/
A,false,N/A
Bill,arn:aws:iam::123456789012:user/Bill,2015-04-15T18:27:44+00:00,false,N/A,N/
A,N/A,false,false,N/A,false,N/A,false,2015-04-20T20:00:12+00:00,false,N/A
```

- Pour plus de détails sur l'API, reportez-vous GetCredentialReportà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def get_credential_report():
    """
    Gets the most recently generated credentials report about the current
 account.

    :return: The credentials report.
    """
    try:
```

```
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]
```

- Pour plus de détails sur l'API, consultez GetCredentialReportle AWS manuel de référence de l'API SDK for Python (Boto3).

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    oo_result = lo_iam->getcredentialreport( ).
    MESSAGE 'Retrieved credential report.' TYPE 'I'.
  CATCH /aws1/cx_iamcredrptnotpresen00.
    MESSAGE 'Credential report not present.' TYPE 'E'.
  CATCH /aws1/cx_iamcredrptexpiredex.
    MESSAGE 'Credential report expired.' TYPE 'E'.
  CATCH /aws1/cx_iamcredrptnotreadyex.
    MESSAGE 'Credential report not ready.' TYPE 'E'.
  CATCH /aws1/cx_iamservicefailureex.
    MESSAGE 'Service failure when getting credential report.' TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous GetCredentialReportà la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`GetGroup`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `GetGroup`.

CLI

AWS CLI

Pour obtenir un groupe IAM

Cet exemple renvoie les informations sur le groupe IAM `Admins`.

```
aws iam get-group \
    --group-name Admins
```

Sortie :

```
{
    "Group": {
        "Path": "/",
        "CreateDate": "2015-06-16T19:41:48Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "GroupName": "Admins"
    },
    "Users": []
}
```

Pour plus d'informations, consultez Identités IAM (utilisateurs, groupes et rôles) dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetGroupà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie des informations sur le groupe IAM **`Testers`**, y compris une collection de tous les utilisateurs IAM appartenant au groupe.

```
$results = Get-IAMGroup -GroupName "Testers"
$results
```

Sortie :

```
Group                                    IsTruncated           Marker
     Users
-----                                    -----------           ------
     -----
Amazon.IdentityManagement.Model.Group    False
     {Theresa, David}
```

```
$results.Group
```

Sortie :

```
Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : 3RHNZZGQJ7QHMAEXAMPLE1
GroupName  : Testers
Path       : /
```

```
$results.Users
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:user/Theresa
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : 4OSVDDJJTF4XEEXAMPLE2
UserName         : Theresa

Arn              : arn:aws:iam::123456789012:user/David
```

```
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE3
UserName         : David
```

- Pour plus de détails sur l'API, reportez-vous GetGroupà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie des informations sur le groupe IAM **`Testers`**, y compris une collection de tous les utilisateurs IAM appartenant au groupe.

```
$results = Get-IAMGroup -GroupName "Testers"
$results
```

Sortie :

```
Group                                    IsTruncated           Marker
     Users
-----                                    -----------           ------
     -----
Amazon.IdentityManagement.Model.Group    False
     {Theresa, David}
```

```
$results.Group
```

Sortie :

```
Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : 3RHNZZGQJ7QHMAEXAMPLE1
GroupName  : Testers
Path       : /
```

```
$results.Users
```

Sortie :

```
Arn             : arn:aws:iam::123456789012:user/Theresa
```

```
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : 4OSVDDJJTF4XEEXAMPLE2
UserName         : Theresa

Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE3
UserName         : David
```

- Pour plus de détails sur l'API, reportez-vous GetGroupà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`GetGroupPolicy`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `GetGroupPolicy`.

CLI

AWS CLI

Pour obtenir des informations sur une politique attachée à un groupe IAM

La commande `get-group-policy` suivante permet d'obtenir des informations sur la politique spécifiée attachée au groupe nommé `Test-Group`.

```
aws iam get-group-policy \
    --group-name Test-Group \
    --policy-name S3-ReadOnly-Policy
```

Sortie :

```
{
    "GroupName": "Test-Group",
    "PolicyDocument": {
```

```
        "Statement": [
            {
                "Action": [
                    "s3:Get*",
                    "s3:List*"
                ],
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    },
    "PolicyName": "S3-ReadOnly-Policy"
}
```

Pour plus d'informations, consultez Gestion des politiques IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetGroupPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie des informations sur la politique intégrée nommée **`PowerUserAccess-Testers`** pour le groupe **`Testers`**. La propriété **`PolicyDocument`** est encodée sous forme d'URL. Elle est décodée dans cet exemple à l'aide de la méthode .NET **`UrlDecode`**.

```
$results = Get-IAMGroupPolicy -GroupName Testers -PolicyName PowerUserAccess-
Testers
$results
```

Sortie :

```
GroupName     PolicyDocument
 PolicyName
---------     --------------
 ----------
Testers       %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%0A%20...
 PowerUserAccess-Testers
```

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
  ]
}
```

- Pour plus de détails sur l'API, reportez-vous GetGroupPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie des informations sur la politique intégrée nommée **`PowerUserAccess-Testers`** pour le groupe **`Testers`**. La propriété **`PolicyDocument`** est encodée sous forme d'URL. Elle est décodée dans cet exemple à l'aide de la méthode .NET **`UrlDecode`**.

```
$results = Get-IAMGroupPolicy -GroupName Testers -PolicyName PowerUserAccess-
Testers
$results
```

Sortie :

```
GroupName     PolicyDocument
 PolicyName
---------     --------------
 ----------
Testers       %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%0A%20...
 PowerUserAccess-Testers

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
  ]
}
```

- Pour plus de détails sur l'API, reportez-vous GetGroupPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`GetInstanceProfile`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `GetInstanceProfile`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Débuter avec le streaming géré

CLI

AWS CLI

Pour obtenir les informations sur un profil d'instance

La commande `get-instance-profile` suivante permet d'obtenir des informations sur le profil d'instance nommé `ExampleInstanceProfile`.

```
aws iam get-instance-profile \
```

```
    --instance-profile-name ExampleInstanceProfile
```

Sortie :

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AID2MAB8DPLSRHEXAMPLE",
        "Roles": [
            {
                "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
                "RoleId": "AIDGPMS9RO4H3FEXAMPLE",
                "CreateDate": "2013-01-09T06:33:26Z",
                "RoleName": "Test-Role",
                "Path": "/",
                "Arn": "arn:aws:iam::336924118301:role/Test-Role"
            }
        ],
        "CreateDate": "2013-06-12T23:52:02Z",
        "InstanceProfileName": "ExampleInstanceProfile",
        "Path": "/",
        "Arn": "arn:aws:iam::336924118301:instance-profile/
ExampleInstanceProfile"
    }
}
```

Pour plus d'informations, consultez Utilisation de profils d'instance dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetInstanceProfileà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple renvoie les détails du profil d'instance nommé **ec2instancerole** défini dans le AWS compte courant.

```
Get-IAMInstanceProfile -InstanceProfileName ec2instancerole
```

Sortie :

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Pour plus de détails sur l'API, reportez-vous GetInstanceProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple renvoie les détails du profil d'instance nommé **`ec2instancerole`** défini dans le AWS compte courant.

```
Get-IAMInstanceProfile -InstanceProfileName ec2instancerole
```

Sortie :

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Pour plus de détails sur l'API, reportez-vous GetInstanceProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`GetLoginProfile`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `GetLoginProfile`.

CLI

AWS CLI

Pour obtenir des informations sur le mot de passe d'un utilisateur IAM

La commande `get-login-profile` suivante permet d'obtenir des informations sur le mot de passe de l'utilisateur IAM nommé Bob.

```
aws iam get-login-profile \
    --user-name Bob
```

Sortie :

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2012-09-21T23:03:39Z"
    }
}
```

La commande `get-login-profile` peut être utilisée pour vérifier qu'un utilisateur IAM possède un mot de passe. La commande renvoie une erreur `NoSuchEntity` si aucun mot de passe n'est défini pour l'utilisateur.

Vous ne pouvez pas afficher un mot de passe à l'aide de cette commande. Si le mot de passe est perdu, vous pouvez le réinitialiser (`update-login-profile`) pour l'utilisateur. Vous pouvez également supprimer le profil de connexion (`delete-login-profile`) de l'utilisateur, puis en recréer un (`create-login-profile`).

Pour plus d'informations, consultez Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetLoginProfileà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie la date de création du mot de passe et indique si une réinitialisation du mot de passe est requise pour l'utilisateur IAM **David**.

```
Get-IAMLoginProfile -UserName David
```

Sortie :

```
CreateDate                  PasswordResetRequired               UserName
----------                  ---------------------               --------
12/10/2014 3:39:44 PM       False                               David
```

- Pour plus de détails sur l'API, reportez-vous GetLoginProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie la date de création du mot de passe et indique si une réinitialisation du mot de passe est requise pour l'utilisateur IAM **`David`**.

```
Get-IAMLoginProfile -UserName David
```

Sortie :

```
CreateDate                  PasswordResetRequired               UserName
----------                  ---------------------               --------
12/10/2014 3:39:44 PM       False                               David
```

- Pour plus de détails sur l'API, reportez-vous GetLoginProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`GetOpenIdConnectProvider`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `GetOpenIdConnectProvider`.

CLI

AWS CLI

Pour renvoyer des informations sur le fournisseur OpenID Connect spécifié

Cet exemple renvoie les informations sur le fournisseur OpenID Connect dont l'ARN est `arn:aws:iam::123456789012:oidc-provider/server.example.com`.

```
aws iam get-open-id-connect-provider \
```

```
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

Sortie :

```
{
    "Url": "server.example.com"
        "CreateDate": "2015-06-16T19:41:48Z",
        "ThumbprintList": [
        "12345abcdefghijk67890lmnopqrst987example"
        ],
        "ClientIDList": [
        "example-application-ID"
        ]
}
```

Pour plus d'informations, consultez Création de fournisseurs d'identité OpenID Connect (OIDC) du Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetOpenIdConnectProviderà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie des informations sur le fournisseur OpenID Connect dont l'ARN est **arn:aws:iam::123456789012:oidc-provider/accounts.google.com**. La propriété **ClientIDList** est une collection qui contient tous les ID client définis pour ce fournisseur.

```
Get-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/oidc.example.com
```

Sortie :

```
ClientIDList         CreateDate               ThumbprintList
           Url
------------         ----------               --------------
           ---
```

```
{MyOIDCApp}          2/3/2015 3:00:30 PM
 {12345abcdefghijk67890lmnopqrst98765uvwxy}   oidc.example.com
```

- Pour plus de détails sur l'API, reportez-vous GetOpenIdConnectProviderà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie des informations sur le fournisseur OpenID Connect dont l'ARN est **`arn:aws:iam::123456789012:oidc-provider/accounts.google.com`**. La propriété **`ClientIDList`** est une collection qui contient tous les ID client définis pour ce fournisseur.

```
Get-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/oidc.example.com
```

Sortie :

```
ClientIDList         CreateDate               ThumbprintList
           Url
------------         ----------               --------------
           ---
{MyOIDCApp}          2/3/2015 3:00:30 PM
 {12345abcdefghijk67890lmnopqrst98765uvwxy}   oidc.example.com
```

- Pour plus de détails sur l'API, reportez-vous GetOpenIdConnectProviderà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`GetPolicy`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `GetPolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Débuter avec le streaming géré

- Utiliser l'API IAM Policy Builder

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::getPolicy(const Aws::String &policyArn,
                            const Aws::Client::ClientConfiguration &clientConfig)
 {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetPolicyRequest request;
    request.SetPolicyArn(policyArn);

    auto outcome = iam.GetPolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error getting policy " << policyArn << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const auto &policy = outcome.GetResult().GetPolicy();
        std::cout << "Name: " << policy.GetPolicyName() << std::endl <<
                  "ID: " << policy.GetPolicyId() << std::endl << "Arn: " <<
                  policy.GetArn() << std::endl << "Description: " <<
                  policy.GetDescription() << std::endl << "CreateDate: " <<

 policy.GetCreateDate().ToGmtString(Aws::Utils::DateFormat::ISO_8601)
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour récupérer les informations sur la politique gérée spécifiée

Cet exemple renvoie des détails sur la politique gérée dont l'ARN est `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam get-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Sortie :

```
{
    "Policy": {
        "PolicyName": "MySamplePolicy",
        "CreateDate": "2015-06-17T19:23;32Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "Z27SI6FQMGNQ2EXAMPLE1",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:policy/MySamplePolicy",
        "UpdateDate": "2015-06-17T19:23:32Z"
    }
}
```

Pour plus d'informations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
```

```
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(ctx context.Context, policyArn string)
 (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(ctx, &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS SDK pour Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Obtenez la politique.

```
import { GetPolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 */
export const getPolicy = (policyArn) => {
  const command = new GetPolicyCommand({
    PolicyArn: policyArn,
  });

  return client.send(command);
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  PolicyArn: "arn:aws:iam::aws:policy/AWSLambdaExecute",
```

```
};

iam.getPolicy(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.Policy.Description);
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun getIAMPolicy(policyArnVal: String?) {
    val request =
        GetPolicyRequest {
            policyArn = policyArnVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.getPolicy(request)
        println("Successfully retrieved policy ${response.policy?.policyName}")
    }
}
```

- Pour plus de détails sur l'API, consultez GetPolicyla section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getPolicy($policyArn)
    {
        return $this->customWaiter(function () use ($policyArn) {
            return $this->iamClient->getPolicy(['PolicyArn' => $policyArn]);
        });
    }
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS SDK pour PHP API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie des détails sur la politique gérée dont l'ARN est **`arn:aws:iam::123456789012:policy/MySamplePolicy`**.

```
Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Sortie :

```
Arn              : arn:aws:iam::aws:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
```

```
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 2/6/2015 10:40:08 AM
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie des détails sur la politique gérée dont l'ARN est **`arn:aws:iam::123456789012:policy/MySamplePolicy`**.

```
Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Sortie :

```
Arn              : arn:aws:iam::aws:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 2/6/2015 10:40:08 AM
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement
```

- Pour plus de détails sur l'API, consultez GetPolicyle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
```

```
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS SDK pour Ruby API.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->getpolicy( iv_policyarn = iv_policy_arn ).
        MESSAGE 'Retrieved policy information.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func getPolicy(arn: String) async throws -> IAMClientTypes.Policy {
        let input = GetPolicyInput(
            policyArn: arn
        )
        do {
            let output = try await client.getPolicy(input: input)
            guard let policy = output.policy else {
                throw ServiceHandlerError.noSuchPolicy
            }
            return policy
        } catch {
            print("ERROR: getPolicy:", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`GetPolicyVersion`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `GetPolicyVersion`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Gérer les politiques
- Utiliser l'API IAM Policy Builder

CLI

AWS CLI

Pour récupérer les informations sur la version spécifiée de la politique gérée spécifiée

Cet exemple renvoie le document de politique pour la version v2 de la politique dont l'ARN est `arn:aws:iam::123456789012:policy/MyManagedPolicy`.

```
aws iam get-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Sortie :

```
{
    "PolicyVersion": {
        "Document": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Action": "iam:*",
                    "Resource": "*"
                }
            ]
        },
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2023-04-11T00:22:54+00:00"
    }
}
```

Pour plus d'informations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetPolicyVersionà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie le document de politique pour la version **v2** de la politique dont l'ARN est **arn:aws:iam::123456789012:policy/MyManagedPolicy** Le document de politique contenu dans la propriété **Document** est encodé sous forme d'URL et est décodé dans cet exemple à l'aide de la méthode .NET **UrlDecode**.

```
$results = Get-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy -VersionId v2
$results
```

Sortie :

```
CreateDate             Document
 IsDefaultVersion     VersionId
----------             --------
 ----------------     ---------
2/12/2015 9:39:53 AM   %7B%0A%20%20%22Version%22%3A%20%222012-10...    True
          v2

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
$policy = [System.Web.HttpUtility]::UrlDecode($results.Document)
$policy
{
  "Version":"2012-10-17",
  "Statement":
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
```

```
    }
}
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyVersionà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie le document de politique pour la version **v2** de la politique dont l'ARN est **arn:aws:iam::123456789012:policy/MyManagedPolicy** Le document de politique contenu dans la propriété **Document** est encodé sous forme d'URL et est décodé dans cet exemple à l'aide de la méthode .NET **UrlDecode**.

```
$results = Get-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy -VersionId v2
$results
```

Sortie :

```
CreateDate             Document
 IsDefaultVersion     VersionId
----------             --------
 ----------------     ---------
2/12/2015 9:39:53 AM   %7B%0A%20%20%22Version%22%3A%20%222012-10...    True
           v2

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
$policy = [System.Web.HttpUtility]::UrlDecode($results.Document)
$policy
{
  "Version":"2012-10-17",
  "Statement":
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
}
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyVersionà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement
```

- Pour plus de détails sur l'API, consultez GetPolicyVersionle AWS manuel de référence de l'API SDK for Python (Boto3).

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    oo_result = lo_iam->getpolicyversion(
      iv_policyarn = iv_policy_arn
      iv_versionid = iv_version_id ).
    MESSAGE 'Retrieved policy version information.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Policy or version does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iaminvalidinputex.
    MESSAGE 'Invalid input provided.' TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyVersionà la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`GetRole`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `GetRole`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Configuration de la connectivité des services de conteneurs
- Création d'une tâche de conteneur pour le type de lancement sans serveur

- Création d'un service de conteneur pour les instances de machines virtuelles
- Premiers pas avec les entrepôts de données sans serveur
- Débuter avec la protection des appareils liés à l'Internet des objets
- Commencer à utiliser les magasins de fonctionnalités d'apprentissage automatique
- Commencer à utiliser les clusters Kubernetes gérés
- Débuter avec le streaming géré
- Commencer à utiliser des clusters d'entrepôts de données provisionnés
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
/// <summary>
/// Get information about an IAM role.
/// </summary>
/// <param name="roleName">The name of the IAM role to retrieve information
/// for.</param>
/// <returns>The IAM role that was retrieved.</returns>
public async Task<Role> GetRoleAsync(string roleName)
{
    var response = await _IAMService.GetRoleAsync(new GetRoleRequest
    {
        RoleName = roleName,
    });

    return response.Role;
}
```

- Pour plus de détails sur l'API, reportez-vous GetRoleà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour obtenir les informations sur un rôle IAM

La commande `get-role` suivante permet d'obtenir des informations sur le rôle nommé `Test-Role`.

```
aws iam get-role \
    --role-name Test-Role
```

Sortie :

```
{
    "Role": {
        "Description": "Test Role",
        "AssumeRolePolicyDocument":"<URL-encoded-JSON>",
        "MaxSessionDuration": 3600,
        "RoleId": "AROA1234567890EXAMPLE",
        "CreateDate": "2019-11-13T16:45:56Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "RoleLastUsed": {
            "Region": "us-east-1",
            "LastUsedDate": "2019-11-13T17:14:00Z"
        },
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

La commande affiche la politique d'approbation attachée au rôle. Pour répertorier les politiques des autorisations attachées à un rôle, utilisez la commande `list-role-policies`.

Pour plus d'informations, consultez Création de rôles IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetRoleà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(ctx context.Context, roleName string)
 (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(ctx,
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
```

```
}
```

- Pour plus de détails sur l'API, reportez-vous GetRoleà la section Référence des AWS SDK pour Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Obtenez le rôle.

```
import { GetRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const getRole = (roleName) => {
  const command = new GetRoleCommand({
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous GetRoleà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getRole($roleName)
    {
        return $this->customWaiter(function () use ($roleName) {
            return $this->iamClient->getRole(['RoleName' => $roleName]);
        });
    }
```

- Pour plus de détails sur l'API, reportez-vous GetRoleà la section Référence des AWS SDK pour PHP API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie les détails du **`lamda_exec_role`**. Il inclut le document de politique d'approbation qui spécifie la personne capable d'assumer ce rôle. Le document de politique est encodé sous forme d'URL et peut être décodé à l'aide de la méthode .NET **`UrlDecode`**. Dans cet exemple, tous les espaces blancs ont été supprimés de la politique originale avant qu'elle ne soit chargée dans la politique. Pour consulter les documents de politique d'autorisation qui déterminent ce qu'une personne assumant le rôle peut faire, utilisez **`Get-IAMRolePolicy`** pour les politiques en ligne, et **`Get-IAMPolicyVersion`** pour les politiques gérées attachées.

```
$results = Get-IamRole -RoleName lambda_exec_role
$results | Format-List
```

Sortie :

```
Arn                       : arn:aws:iam::123456789012:role/lambda_exec_role
AssumeRolePolicyDocument  : %7B%22Version%22%3A%222012-10-17%22%2C%22Statement
%22%3A%5B%7B%22Sid%22
                            %3A%22%22%2C%22Effect%22%3A%22Allow%22%2C%22Principal
%22%3A%7B%22Service
                            %22%3A%22lambda.amazonaws.com%22%7D%2C%22Action%22%3A
%22sts%3AAssumeRole
                            %22%7D%5D%7D
CreateDate                : 4/2/2015 9:16:11 AM
Path                      : /
RoleId                    : 2YBIKAIBHNKB4EXAMPLE1
RoleName                  : lambda_exec_role
```

```
$policy = [System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
$policy
```

Sortie :

```
{"Version":"2012-10-17",       "Statement":
[{"Sid":"","Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}
```

- Pour plus de détails sur l'API, reportez-vous GetRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie les détails du **`lamda_exec_role`**. Il inclut le document de politique d'approbation qui spécifie la personne capable d'assumer ce rôle. Le document de politique est encodé sous forme d'URL et peut être décodé à l'aide de la méthode .NET **`UrlDecode`**. Dans cet exemple, tous les espaces blancs ont été supprimés de la politique originale avant qu'elle ne soit chargée dans la politique. Pour consulter les documents de politique d'autorisation qui déterminent ce qu'une personne assumant le rôle peut faire, utilisez **`Get-IAMRolePolicy`** pour les politiques en ligne, et **`Get-IAMPolicyVersion`** pour les politiques gérées attachées.

```
$results = Get-IamRole -RoleName lambda_exec_role
$results | Format-List
```

Sortie :

```
Arn                      : arn:aws:iam::123456789012:role/lambda_exec_role
AssumeRolePolicyDocument : %7B%22Version%22%3A%222012-10-17%22%2C%22Statement
%22%3A%5B%7B%22Sid%22
                           %3A%22%22%2C%22Effect%22%3A%22Allow%22%2C%22Principal
%22%3A%7B%22Service
                           %22%3A%22lambda.amazonaws.com%22%7D%2C%22Action%22%3A
%22sts%3AAssumeRole
                           %22%7D%5D%7D
CreateDate               : 4/2/2015 9:16:11 AM
Path                     : /
RoleId                   : 2YBIKAIBHNKB4EXAMPLE1
RoleName                 : lambda_exec_role
```

```
$policy = [System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
$policy
```

Sortie :

```
{"Version":"2012-10-17",       "Statement":
[{"Sid":"","Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}
```

- Pour plus de détails sur l'API, reportez-vous GetRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def get_role(role_name):
    """
    Gets a role by name.
```

```
    :param role_name: The name of the role to retrieve.
    :return: The specified role.
    """
    try:
        role = iam.Role(role_name)
        role.load()  # calls GetRole to load attributes
        logger.info("Got role with arn %s.", role.arn)
    except ClientError:
        logger.exception("Couldn't get role named %s.", role_name)
        raise
    else:
        return role
```

- Pour plus de détails sur l'API, consultez GetRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# Gets data about a role.
#
# @param name [String] The name of the role to look up.
# @return [Aws::IAM::Role] The retrieved role.
def get_role(name)
  role = @iam_client.get_role({
                                role_name: name
                              }).role
  puts("Got data for role '#{role.role_name}'. Its ARN is '#{role.arn}'.")
rescue Aws::Errors::ServiceError => e
  puts("Couldn't get data for role '#{name}' Here's why:")
```

```
    puts("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end
```

- Pour plus de détails sur l'API, reportez-vous GetRoleà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn get_role(
    client: &iamClient,
    role_name: String,
) -> Result<GetRoleOutput, SdkError<GetRoleError>> {
    let response = client.get_role().role_name(role_name).send().await?;
    Ok(response)
}
```

- Pour plus de détails sur l'API, voir GetRolela section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->getrole( iv_rolename = iv_role_name ).
        MESSAGE 'Retrieved role information.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role does not exist.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous GetRoleà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func getRole(name: String) async throws -> IAMClientTypes.Role {
        let input = GetRoleInput(
            roleName: name
        )
        do {
```

```
            let output = try await client.getRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
            return role
        } catch {
            print("ERROR: getRole:", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous GetRoleà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`GetRolePolicy`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `GetRolePolicy`.

CLI

AWS CLI

Pour obtenir des informations sur une politique attachée à un rôle IAM

La commande `get-role-policy` suivante permet d'obtenir des informations sur la politique spécifiée attachée au rôle nommé `Test-Role`.

```
aws iam get-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

Sortie :

```
{
  "RoleName": "Test-Role",
  "PolicyDocument": {
```

```
      "Statement": [
          {
              "Action": [
                  "s3:ListBucket",
                  "s3:Put*",
                  "s3:Get*",
                  "s3:*MultipartUpload*"
              ],
              "Resource": "*",
              "Effect": "Allow",
              "Sid": "1"
          }
      ]
  }
  "PolicyName": "ExamplePolicy"
}
```

Pour plus d'informations, consultez Création de rôles IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetRolePolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie le document de politique d'autorisations pour la politique nommée **oneClick_lambda_exec_role_policy** qui est intégrée au rôle IAM **lamda_exec_role**. Le document de politique qui en résulte est encodé sous forme d'URL. Elle est décodée dans cet exemple à l'aide de la méthode .NET **UrlDecode**.

```
$results = Get-IAMRolePolicy -RoleName lambda_exec_role -PolicyName
 oneClick_lambda_exec_role_policy
$results
```

Sortie :

```
PolicyDocument                                          PolicyName
              UserName
--------------                                          ----------
              --------
```

```
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...
 oneClick_lambda_exec_role_policy     lambda_exec_role
```

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
```

Sortie :

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:*"
      ],
      "Resource": "arn:aws:logs:us-east-1:555555555555:log-group:/aws/lambda/aws-
example-function:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ]
    }
  ]
}
```

- Pour plus de détails sur l'API, reportez-vous GetRolePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie le document de politique d'autorisations pour la politique nommée **`oneClick_lambda_exec_role_policy`** qui est intégrée au rôle IAM **`lamda_exec_role`**. Le document de politique qui en résulte est encodé sous forme d'URL. Elle est décodée dans cet exemple à l'aide de la méthode .NET **`UrlDecode`**.

```
$results = Get-IAMRolePolicy -RoleName lambda_exec_role -PolicyName
 oneClick_lambda_exec_role_policy
$results
```

Sortie :

```
PolicyDocument                                              PolicyName
              UserName
--------------                                              ----------
              --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...
 oneClick_lambda_exec_role_policy     lambda_exec_role
```

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
```

Sortie :

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:*"
      ],
      "Resource": "arn:aws:logs:us-east-1:555555555555:log-group:/aws/lambda/aws-
example-function:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ]
    }
  ]
}
```

- Pour plus de détails sur l'API, reportez-vous GetRolePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`GetSamlProvider`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `GetSamlProvider`.

CLI

AWS CLI

Pour récupérer le métadocument du fournisseur SAML

Cet exemple extrait les informations relatives au fournisseur SAML 2.0 dont l'ARM est le nom `arn:aws:iam::123456789012:saml-provider/SAMLADFS`. La réponse inclut le document de métadonnées que vous avez obtenu du fournisseur d'identité pour créer l'entité du fournisseur AWS SAML ainsi que les dates de création et d'expiration.

```
aws iam get-saml-provider \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

Sortie :

```
{
    "SAMLMetadataDocument": "...SAMLMetadataDocument-XML...",
    "CreateDate": "2017-03-06T22:29:46+00:00",
    "ValidUntil": "2117-03-06T22:29:46.433000+00:00",
    "Tags": [
        {
            "Key": "DeptID",
            "Value": "123456"
        },
        {
            "Key": "Department",
            "Value": "Accounting"
        }
    ]
```

```
}
```

Pour plus d'informations, consultez Création de fournisseurs d'identité SAML IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetSamlProviderà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple récupère les informations relatives au fournisseur SAML 2.0 dont l'ARM est arn:aws:iam : :123456789012:saml-. provider/SAMLADFS La réponse inclut le document de métadonnées que vous avez obtenu du fournisseur d'identité pour créer l'entité du fournisseur AWS SAML ainsi que les dates de création et d'expiration.

```
Get-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS
```

Sortie :

```
CreateDate                 SAMLMetadataDocument
        ValidUntil
----------                 --------------------
        ----------
12/23/2014 12:16:55 PM    <EntityDescriptor ID="_12345678-1234-5678-9012-
example1...    12/23/2114 12:16:54 PM
```

- Pour plus de détails sur l'API, reportez-vous GetSamlProviderà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple récupère les informations relatives au fournisseur SAML 2.0 dont l'ARM est arn:aws:iam : :123456789012:saml-. provider/SAMLADFS La réponse inclut le document de métadonnées que vous avez obtenu du fournisseur d'identité pour créer l'entité du fournisseur AWS SAML ainsi que les dates de création et d'expiration.

```
Get-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS
```

Sortie :

```
CreateDate                 SAMLMetadataDocument
        ValidUntil
----------                 --------------------
        ----------
12/23/2014 12:16:55 PM    <EntityDescriptor ID="_12345678-1234-5678-9012-
example1...    12/23/2114 12:16:54 PM
```

- Pour plus de détails sur l'API, reportez-vous GetSamlProviderà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`GetServerCertificate`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `GetServerCertificate`.

C++

Kit de développement logiciel (SDK) for C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::getServerCertificate(const Aws::String &certificateName,
                                       const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetServerCertificateRequest request;
    request.SetServerCertificateName(certificateName);

    auto outcome = iam.GetServerCertificate(request);
    bool result = true;
    if (!outcome.IsSuccess()) {
```

```
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error getting server certificate " << certificateName
 <<
                      ": " << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << certificateName
                      << "' not found." << std::endl;
        }
    }
    else {
        const auto &certificate = outcome.GetResult().GetServerCertificate();
        std::cout << "Name: " <<

 certificate.GetServerCertificateMetadata().GetServerCertificateName()
                  << std::endl << "Body: " << certificate.GetCertificateBody() <<
                  std::endl << "Chain: " << certificate.GetCertificateChain() <<
                  std::endl;
    }

    return result;
}
```

- Pour plus de détails sur l'API, reportez-vous GetServerCertificateà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour obtenir des informations sur un certificat de serveur associé à votre AWS compte

La `get-server-certificate` commande suivante permet de récupérer tous les détails relatifs au certificat de serveur spécifié dans votre AWS compte.

```
aws iam get-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

Sortie :

```
{
    "ServerCertificate": {
        "ServerCertificateMetadata": {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        "CertificateBody": "-----BEGIN CERTIFICATE-----
            MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
            VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
            b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
            BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
            MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
            VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
            b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
            YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
            21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
            rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
            Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
            nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
            FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
            NYiytVbZPQUQ5Yaxu2jXnimvrszlaEXAMPLE=-----END CERTIFICATE-----",
        "CertificateChain": "-----BEGIN CERTIFICATE-----\nMIICiTCCAfICCQD6md
            7oRw0uXOjANBgkqhkiG9w0BAqQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgT
            AldBMRAwDgYDVQQHEwdTZWF0drGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAs
            TC0lBTSBDb25zb2xlMRIwEAYDVsQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQ
            jb20wHhcNMTEwNDI1MjA0NTIxWhtcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBh
            MCVVMxCzAJBgNVBAgTAldBMRAwDgsYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBb
            WF6b24xFDASBgNVBAsTC0lBTSBDb2d5zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMx
            HzAdBgkqhkiG9w0BCQEWEG5vb25lQGfFtYXpvbi5jb20wgZ8wDQYJKoZIhvcNAQE
            BBQADgY0AMIGJAoGBAMaK0dn+a4GmWIgWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVI
            k60CpiwsZ3G93vUEIO3IyNoH/f0wYK8mh9TrDHudUZg3qX4waLG5M43q7Wgc/MbQ
            ITxOUSQv7c7ugFFDzQGBzZswY6786m86gjpEIbb3OhjZnzcvQAaRHhdlQWIMm2nr
            AgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCku4nUhVVxYUntneD9+h8Mg9q6q+auN
            KyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FlkbFFBjvSfpJIlJ00zbhNYS5f6Guo
            EDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjS;TbNYiytVbZPQUQ5Yaxu2jXnimvw
            3rrszlaEWEG5vb25lQGFtsYXpvbiEXAMPLE=\n-----END CERTIFICATE-----"
    }
}
```

Pour répertorier les certificats de serveur disponibles dans votre AWS compte, utilisez la `list-server-certificates` commande.

Pour plus d'informations, consultez Gestion des certificats de serveur dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetServerCertificateà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Obtenez un certificat de serveur.

```
import { GetServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 * @returns
 */
export const getServerCertificate = async (certName) => {
  const command = new GetServerCertificateCommand({
    ServerCertificateName: certName,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.

- Pour plus de détails sur l'API, reportez-vous GetServerCertificateà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.getServerCertificate(
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
);
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous GetServerCertificateà la section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple extrait les détails du certificat de serveur nommé **MyServerCertificate**. Vous trouverez les détails du certificat dans les propriétés **CertificateBody** et **ServerCertificateMetadata**.

```
$result = Get-IAMServerCertificate -ServerCertificateName MyServerCertificate
$result | format-list
```

Sortie :

```
CertificateBody           : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt

 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
                           21uUSfwfEvySWtC2XADZ4nB
+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
                           rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                           nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                           NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                           -----END CERTIFICATE-----
CertificateChain          :
```

```
ServerCertificateMetadata :
 Amazon.IdentityManagement.Model.ServerCertificateMetadata
```

```
$result.ServerCertificateMetadata
```

Sortie :

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Pour plus de détails sur l'API, reportez-vous GetServerCertificateà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple extrait les détails du certificat de serveur nommé **MyServerCertificate**. Vous trouverez les détails du certificat dans les propriétés **CertificateBody** et **ServerCertificateMetadata**.

```
$result = Get-IAMServerCertificate -ServerCertificateName MyServerCertificate
$result | format-list
```

Sortie :

```
CertificateBody           : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
```

```
 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt

 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
                            21uUSfwfEvySWtC2XADZ4nB
+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
                            rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                            nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                            NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                            -----END CERTIFICATE-----
CertificateChain          :
ServerCertificateMetadata :
 Amazon.IdentityManagement.Model.ServerCertificateMetadata
```

```
$result.ServerCertificateMetadata
```

Sortie :

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Pour plus de détails sur l'API, reportez-vous GetServerCertificateà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`GetServiceLastAccessedDetails`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `GetServiceLastAccessedDetails`.

CLI

AWS CLI

Pour récupérer un rapport d'accès aux services

L'exemple `get-service-last-accessed-details` suivant récupère un rapport généré précédemment qui répertorie les services auxquels accèdent les entités IAM. Pour générer un rapport, utilisez la commande `generate-service-last-accessed-details`.

```
aws iam get-service-last-accessed-details \
    --job-id 2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc
```

Sortie :

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-10-01T03:50:35.929Z",
    "ServicesLastAccessed": [
        ...
        {
            "ServiceName": "AWS Lambda",
            "LastAuthenticated": "2019-09-30T23:02:00Z",
            "ServiceNamespace": "lambda",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/admin",
            "TotalAuthenticatedEntities": 6
        },
    ]
}
```

Pour plus d'informations, voir Affiner les autorisations lors de AWS l'utilisation des dernières informations consultées dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir GetServiceLastAccessedDetailsla section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple fournit des détails sur le dernier service auquel a accédé l'entité IAM (utilisateur, groupe, rôle ou politique) associée à l'appel de la requête.

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

Sortie :

```
f0b7a819-eab0-929b-dc26-ca598911cb9f
```

```
Get-IAMServiceLastAccessedDetail -JobId f0b7a819-eab0-929b-dc26-ca598911cb9f
```

- Pour plus de détails sur l'API, reportez-vous GetServiceLastAccessedDetailsà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple fournit des détails sur le dernier service auquel a accédé l'entité IAM (utilisateur, groupe, rôle ou politique) associée à l'appel de la requête.

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

Sortie :

```
f0b7a819-eab0-929b-dc26-ca598911cb9f
```

```
Get-IAMServiceLastAccessedDetail -JobId f0b7a819-eab0-929b-dc26-ca598911cb9f
```

- Pour plus de détails sur l'API, reportez-vous GetServiceLastAccessedDetailsà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`GetServiceLastAccessedDetailsWithEntities`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `GetServiceLastAccessedDetailsWithEntities`.

CLI

AWS CLI

Pour récupérer un rapport d'accès aux services contenant les détails d'un service

L'exemple `get-service-last-accessed-details-with-entities` suivant récupère un rapport contenant des informations détaillées sur les utilisateurs IAM et les autres entités ayant accédé au service spécifié. Pour générer un rapport, utilisez la commande `generate-service-last-accessed-details`. Pour obtenir une liste des services auxquels on accède à l'aide d'espaces de noms, utilisez `get-service-last-accessed-details`.

```
aws iam get-service-last-accessed-details-with-entities \
    --job-id 78b6c2ba-d09e-6xmp-7039-ecde30b26916 \
    --service-namespace lambda
```

Sortie :

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-10-01T03:55:41.756Z",
    "JobCompletionDate": "2019-10-01T03:55:42.533Z",
    "EntityDetailsList": [
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/admin",
                "Name": "admin",
                "Type": "USER",
                "Id": "AIDAIO2XMPLENQEXAMPLE",
                "Path": "/"
            },
            "LastAuthenticated": "2019-09-30T23:02:00Z"
        },
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/developer",
                "Name": "developer",
                "Type": "USER",
```

```
                "Id": "AIDAIBEYXMPL2YEXAMPLE",
                "Path": "/"
            },
            "LastAuthenticated": "2019-09-16T19:34:00Z"
        }
    ]
}
```

Pour plus d'informations, voir Affiner les autorisations lors de AWS l'utilisation des dernières informations consultées dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir GetServiceLastAccessedDetailsWithEntitiesla section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple fournit l'horodatage du dernier accès au service dans la requête par cette entité IAM respective.

```
$results = Get-IAMServiceLastAccessedDetailWithEntity -JobId f0b7a819-eab0-929b-
dc26-ca598911cb9f -ServiceNamespace ec2
$results
```

Sortie :

```
EntityDetailsList : {Amazon.IdentityManagement.Model.EntityDetails}
Error             :
IsTruncated       : False
JobCompletionDate : 12/29/19 11:19:31 AM
JobCreationDate   : 12/29/19 11:19:31 AM
JobStatus         : COMPLETED
Marker            :
```

```
$results.EntityDetailsList
```

Sortie :

```
EntityInfo                                LastAuthenticated
----------                                -----------------
```

```
Amazon.IdentityManagement.Model.EntityInfo 11/16/19 3:47:00 PM
```

```
$results.EntityInfo
```

Sortie :

```
Arn  : arn:aws:iam::123456789012:user/TestUser
Id   : AIDA4NBK5CXF5TZHU1234
Name : TestUser
Path : /
Type : USER
```

- Pour plus de détails sur l'API, reportez-vous GetServiceLastAccessedDetailsWithEntitiesà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple fournit l'horodatage du dernier accès au service dans la requête par cette entité IAM respective.

```
$results = Get-IAMServiceLastAccessedDetailWithEntity -JobId f0b7a819-eab0-929b-
dc26-ca598911cb9f -ServiceNamespace ec2
$results
```

Sortie :

```
EntityDetailsList : {Amazon.IdentityManagement.Model.EntityDetails}
Error             :
IsTruncated       : False
JobCompletionDate : 12/29/19 11:19:31 AM
JobCreationDate   : 12/29/19 11:19:31 AM
JobStatus         : COMPLETED
Marker            :
```

```
$results.EntityDetailsList
```

Sortie :

```
EntityInfo                                LastAuthenticated
----------                                -----------------
Amazon.IdentityManagement.Model.EntityInfo 11/16/19 3:47:00 PM
```

```
$results.EntityInfo
```

Sortie :

```
Arn  : arn:aws:iam::123456789012:user/TestUser
Id   : AIDA4NBK5CXF5TZHU1234
Name : TestUser
Path : /
Type : USER
```

- Pour plus de détails sur l'API, reportez-vous GetServiceLastAccessedDetailsWithEntitiesà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`GetServiceLinkedRoleDeletionStatus`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `GetServiceLinkedRoleDeletionStatus`.

CLI

AWS CLI

Pour vérifier le statut d'une requête de suppression d'un rôle lié à un service

L'exemple `get-service-linked-role-deletion-status` suivant affiche le statut d'une requête précédente de suppression d'un rôle lié à un service. L'opération de suppression s'effectue de manière asynchrone. Lorsque vous effectuez la requête, vous obtenez une valeur `DeletionTaskId` que vous fournissez en tant que paramètre de cette commande.

```
aws iam get-service-linked-role-deletion-status \
    --deletion-task-id task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE
```

Sortie :

```
{
```

```
"Status": "SUCCEEDED"
}
```

Pour plus d'informations, consultez Utilisation des rôles liés à un service dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetServiceLinkedRoleDeletionStatusà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import {
  GetServiceLinkedRoleDeletionStatusCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} deletionTaskId
 */
export const getServiceLinkedRoleDeletionStatus = (deletionTaskId) => {
  const command = new GetServiceLinkedRoleDeletionStatusCommand({
    DeletionTaskId: deletionTaskId,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, voir GetServiceLinkedRoleDeletionStatusla section Référence des AWS SDK pour JavaScript API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`GetUser`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `GetUser`.

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }
```

- Pour plus de détails sur l'API, reportez-vous GetUserà la section Référence des AWS SDK pour .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
##############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)
```

```
  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}
```

- Pour plus de détails sur l'API, reportez-vous GetUserà la section Référence des AWS CLI commandes.

CLI

AWS CLI

Pour obtenir les informations sur un utilisateur IAM

La commande `get-user` suivante permet d'obtenir des informations sur l'utilisateur IAM nommé `Paulo`.

```
aws iam get-user \
    --user-name Paulo
```

Sortie :

```
{
    "User": {
        "UserName": "Paulo",
        "Path": "/",
        "CreateDate": "2019-09-21T23:03:13Z",
        "UserId": "AIDA123456789EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Paulo"
    }
```

```
}
```

Pour plus d'informations, consultez Gestion des utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetUserà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(ctx context.Context, userName string)
 (*types.User, error) {
```

```
 var user *types.User
 result, err := wrapper.IamClient.GetUser(ctx, &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}
```

- Pour plus de détails sur l'API, reportez-vous GetUserà la section Référence des AWS SDK pour Go API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple permet de récupérer des informations sur l'utilisateur nommé **David**.

```
Get-IAMUser -UserName David
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE1
UserName         : David
```

Exemple 2 : cet exemple récupère les détails de l'utilisateur IAM actuellement connecté.

```
Get-IAMUser
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE2
UserName         : Bob
```

- Pour plus de détails sur l'API, reportez-vous GetUserà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple permet de récupérer des informations sur l'utilisateur nommé **David**.

```
Get-IAMUser -UserName David
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE1
UserName         : David
```

Exemple 2 : cet exemple récupère les détails de l'utilisateur IAM actuellement connecté.

```
Get-IAMUser
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 10/16/2014 9:03:09 AM
```

```
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE2
UserName         : Bob
```

- Pour plus de détails sur l'API, reportez-vous GetUserà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Retrieves a user's details
 #
 # @param user_name [String] The name of the user to retrieve
 # @return [Aws::IAM::Types::User, nil] The user object if found, or nil if an
error occurred
 def get_user(user_name)
   response = @iam_client.get_user(user_name: user_name)
   response.user
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error("User '#{user_name}' not found.")
   nil
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error retrieving user '#{user_name}': #{e.message}")
   nil
 end
```

- Pour plus de détails sur l'API, reportez-vous GetUserà la section Référence des AWS SDK pour Ruby API.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func getUser(name: String? = nil) async throws -> IAMClientTypes.User
 {
        let input = GetUserInput(
            userName: name
        )
        do {
            let output = try await iamClient.getUser(input: input)
            guard let user = output.user else {
                throw ServiceHandlerError.noSuchUser
            }
            return user
        } catch {
            print("ERROR: getUser:", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous GetUserà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`GetUserPolicy`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `GetUserPolicy`.

CLI

AWS CLI

Pour répertorier les détails de la politique d'un utilisateur IAM

La commande `get-user-policy` suivante répertorie les détails de la politique spécifiée qui est attachée à l'utilisateur IAM nommé Bob.

```
aws iam get-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

Sortie :

```
{
    "UserName": "Bob",
    "PolicyName": "ExamplePolicy",
    "PolicyDocument": {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Action": "*",
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    }
}
```

Pour obtenir une liste des politiques d'un utilisateur IAM, utilisez la commande `list-user-policies`.

Pour plus d'informations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetUserPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple extrait les détails de la politique intégrée nommée **`Davids_IAM_Admin_Policy`** intégrée dans le nom de l'utilisateur IAM **`David`**. Le document de politique est encodé sous forme d'URL.

```
$results = Get-IAMUserPolicy -PolicyName Davids_IAM_Admin_Policy -UserName David
$results
```

Sortie :

```
PolicyDocument                                                  PolicyName
       UserName
--------------                                                  ----------
       --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...    Davids_IAM_Admin_Policy
       David

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:GetUser",
        "iam:ListUsers"
      ],
      "Resource": [
        "arn:aws:iam::111122223333:user/*"
      ]
    }
  ]
}
```

- Pour plus de détails sur l'API, reportez-vous GetUserPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple extrait les détails de la politique intégrée nommée **`Davids_IAM_Admin_Policy`** intégrée dans le nom de l'utilisateur IAM **`David`**. Le document de politique est encodé sous forme d'URL.

```
$results = Get-IAMUserPolicy -PolicyName Davids_IAM_Admin_Policy -UserName David
$results
```

Sortie :

```
PolicyDocument                                                  PolicyName
       UserName
--------------                                                  ----------
       --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...    Davids_IAM_Admin_Policy
       David

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:GetUser",
        "iam:ListUsers"
      ],
      "Resource": [
        "arn:aws:iam::111122223333:user/*"
      ]
    }
  ]
}
```

- Pour plus de détails sur l'API, reportez-vous GetUserPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`ListAccessKeys`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `ListAccessKeys`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer des clés d'accès

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_list_access_keys
#
# This function lists the access keys for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#
```

```
# Returns:
#       access_key_ids
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_list_access_keys() {

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_access_keys"
    echo "Lists the AWS Identity and Access Management (IAM) access key IDs for
 the specified user."
    echo "  -u user_name   The name of the IAM user."
    echo ""
  }

  local user_name response
  local option OPTARG # Required to use getopts command in a function.
  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam list-access-keys \
    --user-name "$user_name" \
```

```
    --output text \
    --query 'AccessKeyMetadata[].AccessKeyId')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-access-keys operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous ListAccessKeysà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::listAccessKeys(const Aws::String &userName,
                                 const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccessKeysRequest request;
    request.SetUserName(userName);

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListAccessKeys(request);
```

```
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list access keys for user " << userName
                      << ": " << outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(32) << "UserName" <<
                      std::setw(30) << "KeyID" << std::setw(20) << "Status" <<
                      std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &keys = outcome.GetResult().GetAccessKeyMetadata();
        const Aws::String DATE_FORMAT = "%Y-%m-%d";

        for (const auto &key: keys) {
            Aws::String statusString =
                    Aws::IAM::Model::StatusTypeMapper::GetNameForStatusType(
                            key.GetStatus());
            std::cout << std::left << std::setw(32) << key.GetUserName() <<
                      std::setw(30) << key.GetAccessKeyId() << std::setw(20) <<
                      statusString << std::setw(20) <<
                      key.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
 std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Pour plus de détails sur l'API, reportez-vous ListAccessKeysà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour répertorier les ID de clés d'accès d'un utilisateur IAM

La commande `list-access-keys` suivante répertorie les ID des clés d'accès de l'utilisateur IAM nommé Bob.

```
aws iam list-access-keys \
    --user-name Bob
```

Sortie :

```
{
    "AccessKeyMetadata": [
        {
            "UserName": "Bob",
            "Status": "Active",
            "CreateDate": "2013-06-04T18:17:34Z",
            "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
        },
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CreateDate": "2013-06-06T20:42:26Z",
            "AccessKeyId": "AKIAI44QH8DHBEXAMPLE"
        }
    ]
}
```

Vous ne pouvez pas répertorier les clés d'accès secrètes des utilisateurs IAM. Si les clés d'accès secrètes sont perdues, vous devez créer de nouvelles clés d'accès à l'aide de la commande `create-access-keys`.

Pour plus d'informations, consultez Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListAccessKeysà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(ctx context.Context, userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(ctx, &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
```

```
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Pour plus de détails sur l'API, reportez-vous ListAccessKeysà la section Référence des AWS SDK pour Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.AccessKeyMetadata;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccessKeysRequest;
import software.amazon.awssdk.services.iam.model.ListAccessKeysResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListAccessKeys {
    public static void main(String[] args) {
        final String usage = """

                Usage:
```

```
                    <userName>\s

                Where:
                    userName - The name of the user for which access keys are
retrieved.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listKeys(iam, userName);
        System.out.println("Done");
        iam.close();
    }

    public static void listKeys(IamClient iam, String userName) {
        try {
            boolean done = false;
            String newMarker = null;

            while (!done) {
                ListAccessKeysResponse response;

                if (newMarker == null) {
                    ListAccessKeysRequest request =
ListAccessKeysRequest.builder()
                            .userName(userName)
                            .build();

                    response = iam.listAccessKeys(request);

                } else {
                    ListAccessKeysRequest request =
ListAccessKeysRequest.builder()
                            .userName(userName)
                            .marker(newMarker)
```

```
                            .build();

                    response = iam.listAccessKeys(request);
                }

                for (AccessKeyMetadata metadata : response.accessKeyMetadata()) {
                    System.out.format("Retrieved access key %s",
 metadata.accessKeyId());
                }

                if (!response.isTruncated()) {
                    done = true;
                } else {
                    newMarker = response.marker();
                }
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListAccessKeysà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les clés d'accès.

```
import { ListAccessKeysCommand, IAMClient } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} userName
 */
export async function* listAccessKeys(userName) {
  const command = new ListAccessKeysCommand({
    MaxItems: 5,
    UserName: userName,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListAccessKeysCommandOutput |
 undefined}
   */
  let response = await client.send(command);

  while (response?.AccessKeyMetadata?.length) {
    for (const key of response.AccessKeyMetadata) {
      yield key;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAccessKeysCommand({
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous ListAccessKeysà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 5,
  UserName: "IAM_USER_NAME",
};

iam.listAccessKeys(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous ListAccessKeysà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun listKeys(userNameVal: String?) {
    val request =
        ListAccessKeysRequest {
            userName = userNameVal
        }
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccessKeys(request)
        response.accessKeyMetadata?.forEach { md ->
            println("Retrieved access key ${md.accessKeyId}")
        }
    }
}
```

- Pour plus de détails sur l'API, consultez ListAccessKeysla section AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cette commande répertorie les clés d'accès de l'utilisateur IAM nommé **Bob**. Notez que vous ne pouvez pas répertorier les clés d'accès secrètes des utilisateurs IAM. Si les clés d'accès secrètes sont perdues, vous devez générer de nouvelles clés d'accès à l'applet de commande **New-IAMAccessKey**.

```
Get-IAMAccessKey -UserName "Bob"
```

Sortie :

```
AccessKeyId               CreateDate                 Status
 UserName
-----------               ----------                 ------
 --------
AKIAIOSFODNN7EXAMPLE      12/3/2014 10:53:41 AM      Active             Bob
AKIAI44QH8DHBEXAMPLE      6/6/2013 8:42:26 PM        Inactive           Bob
```

- Pour plus de détails sur l'API, reportez-vous ListAccessKeysà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cette commande répertorie les clés d'accès de l'utilisateur IAM nommé **Bob**. Notez que vous ne pouvez pas répertorier les clés d'accès secrètes des utilisateurs IAM. Si les clés d'accès secrètes sont perdues, vous devez générer de nouvelles clés d'accès à l'applet de commande **New-IAMAccessKey**.

```
Get-IAMAccessKey -UserName "Bob"
```

Sortie :

```
AccessKeyId               CreateDate                 Status
 UserName
-----------               ----------                 ------
 --------
AKIAIOSFODNN7EXAMPLE      12/3/2014 10:53:41 AM      Active             Bob
AKIAI44QH8DHBEXAMPLE      6/6/2013 8:42:26 PM        Inactive           Bob
```

- Pour plus de détails sur l'API, reportez-vous ListAccessKeysà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_keys(user_name):
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
    else:
        return keys
```

- Pour plus de détails sur l'API, consultez ListAccessKeysle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, désactive et supprime les clés d'accès.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'AccessKeyManager'
  end
```

```
 # Lists access keys for a user
 #
 # @param user_name [String] The name of the user.
 def list_access_keys(user_name)
   response = @iam_client.list_access_keys(user_name: user_name)
   if response.access_key_metadata.empty?
     @logger.info("No access keys found for user '#{user_name}'.")
   else
     response.access_key_metadata.map(&:access_key_id)
   end
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
   []
 rescue StandardError => e
   @logger.error("Error listing access keys: #{e.message}")
   []
 end

 # Creates an access key for a user
 #
 # @param user_name [String] The name of the user.
 # @return [Boolean]
 def create_access_key(user_name)
   response = @iam_client.create_access_key(user_name: user_name)
   access_key = response.access_key
   @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
   access_key
 rescue Aws::IAM::Errors::LimitExceeded
   @logger.error('Error creating access key: limit exceeded. Cannot create
more.')
   nil
 rescue StandardError => e
   @logger.error("Error creating access key: #{e.message}")
   nil
 end

 # Deactivates an access key
 #
 # @param user_name [String] The name of the user.
 # @param access_key_id [String] The ID for the access key.
 # @return [Boolean]
 def deactivate_access_key(user_name, access_key_id)
```

```
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: 'Inactive'
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, reportez-vous ListAccessKeysà la section Référence des AWS SDK pour Ruby API.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->listaccesskeys(
          iv_username = iv_user_name ).
        MESSAGE 'Retrieved access key list.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User does not exist.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous ListAccessKeysà la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`ListAccountAliases`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `ListAccountAliases`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer votre compte

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool
AwsDoc::IAM::listAccountAliases(const Aws::Client::ClientConfiguration
 &clientConfig) {
```

```
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccountAliasesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListAccountAliases(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list account aliases: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        const auto &aliases = outcome.GetResult().GetAccountAliases();
        if (!header) {
            if (aliases.size() == 0) {
                std::cout << "Account has no aliases" << std::endl;
                break;
            }
            std::cout << std::left << std::setw(32) << "Alias" << std::endl;
            header = true;
        }

        for (const auto &alias: aliases) {
            std::cout << std::left << std::setw(32) << alias << std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Pour plus de détails sur l'API, reportez-vous ListAccountAliasesà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour répertorier des alias de compte

La commande `list-account-aliases` suivante répertorie les alias du compte actuel.

```
aws iam list-account-aliases
```

Sortie :

```
{
    "AccountAliases": [
    "mycompany"
    ]
}
```

Pour plus d'informations, consultez la section Votre identifiant de AWS compte et son alias dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListAccountAliasesà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccountAliasesResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
```

```
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListAccountAliases {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listAliases(iam);
        System.out.println("Done");
        iam.close();
    }

    public static void listAliases(IamClient iam) {
        try {
            ListAccountAliasesResponse response = iam.listAccountAliases();
            for (String alias : response.accountAliases()) {
                System.out.printf("Retrieved account alias %s", alias);
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListAccountAliasesà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les alias de compte.

```
import { ListAccountAliasesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 */
export async function* listAccountAliases() {
  const command = new ListAccountAliasesCommand({ MaxItems: 5 });

  let response = await client.send(command);

  while (response.AccountAliases?.length) {
    for (const alias of response.AccountAliases) {
      yield alias;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAccountAliasesCommand({
          Marker: response.Marker,
          MaxItems: 5,
        }),
      );
    } else {
      break;
    }
  }
```

```
}
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous ListAccountAliasesà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.listAccountAliases({ MaxItems: 10 }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous ListAccountAliasesà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun listAliases() {
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccountAliases(ListAccountAliasesRequest {})
        response.accountAliases?.forEach { alias ->
            println("Retrieved account alias $alias")
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListAccountAliasesà la section AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cette commande renvoie l'alias de compte pour l' Compte AWS.

```
Get-IAMAccountAlias
```

Sortie :

```
ExampleCo
```

- Pour plus de détails sur l'API, reportez-vous ListAccountAliasesà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cette commande renvoie l'alias de compte pour l' Compte AWS.

```
Get-IAMAccountAlias
```

Sortie :

```
ExampleCo
```

- Pour plus de détails sur l'API, reportez-vous ListAccountAliasesà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_aliases():
    """
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
        else:
            logger.info("Got no aliases for your account.")
    except ClientError:
        logger.exception("Couldn't list aliases for your account.")
        raise
    else:
        return response["AccountAliases"]
```

- Pour plus de détails sur l'API, consultez ListAccountAliasesle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Listez, créez et supprimez les alias de compte.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info('Account aliases are:')
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info('No account aliases found.')
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end
```

```
  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, reportez-vous ListAccountAliasesà la section Référence des AWS SDK pour Ruby API.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->listaccountaliases( ).
```

```
        MESSAGE 'Retrieved account alias list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing account aliases.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous ListAccountAliasesà la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`ListAttachedGroupPolicies`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `ListAttachedGroupPolicies`.

CLI

AWS CLI

Pour répertorier toutes les politiques gérées attachées au groupe spécifié

Cet exemple renvoie les noms et les ARN des politiques gérées associées au groupe IAM nommé `Admins` dans le AWS compte.

```
aws iam list-attached-group-policies \
    --group-name Admins
```

Sortie :

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
```

```
    ],
    "IsTruncated": false
}
```

Pour plus d'informations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListAttachedGroupPoliciesà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cette commande renvoie les noms et les ARN des politiques gérées associées au groupe IAM nommé **Admins** dans le AWS compte. Pour consulter la liste des politiques intégrées au groupe, utilisez la commande **Get-IAMGroupPolicyList**.

```
Get-IAMAttachedGroupPolicyList -GroupName "Admins"
```

Sortie :

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
arn:aws:iam::aws:policy/AdministratorAccess               AdministratorAccess
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedGroupPoliciesà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cette commande renvoie les noms et les ARN des politiques gérées associées au groupe IAM nommé **Admins** dans le AWS compte. Pour consulter la liste des politiques intégrées au groupe, utilisez la commande **Get-IAMGroupPolicyList**.

```
Get-IAMAttachedGroupPolicyList -GroupName "Admins"
```

Sortie :

```
PolicyArn                                                  PolicyName
---------                                                  ----------
arn:aws:iam::aws:policy/SecurityAudit                      SecurityAudit
arn:aws:iam::aws:policy/AdministratorAccess                AdministratorAccess
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedGroupPoliciesà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`ListAttachedRolePolicies`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `ListAttachedRolePolicies`.

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
```

```
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }
```

- Pour plus de détails sur l'API, voir ListAttachedRolePoliciesla section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour répertorier toutes les politiques gérées qui sont attachées au rôle spécifié

Cette commande renvoie les noms et les ARN des politiques gérées associées au rôle IAM nommé `SecurityAuditRole` dans le AWS compte.

```
aws iam list-attached-role-policies \
    --role-name SecurityAuditRole
```

Sortie :

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Pour plus d'informations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir ListAttachedRolePoliciesla section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(ctx context.Context, roleName
 string) ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(ctx,
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
```

```
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}
```

- Pour plus de détails sur l'API, voir ListAttachedRolePoliciesla section Référence des AWS SDK pour Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les politiques qui sont attachées à un rôle.

```
import {
  ListAttachedRolePoliciesCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 * @param {string} roleName
 */
export async function* listAttachedRolePolicies(roleName) {
  const command = new ListAttachedRolePoliciesCommand({
    RoleName: roleName,
  });
```

```
  let response = await client.send(command);

  while (response.AttachedPolicies?.length) {
    for (const policy of response.AttachedPolicies) {
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAttachedRolePoliciesCommand({
          RoleName: roleName,
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Pour plus de détails sur l'API, voir ListAttachedRolePoliciesla section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listAttachedRolePolicies($roleName, $pathPrefix = "", $marker
 = "", $maxItems = 0)
    {
        $listAttachRolePoliciesArguments = ['RoleName' => $roleName];
```

```
        if ($pathPrefix) {
            $listAttachRolePoliciesArguments['PathPrefix'] = $pathPrefix;
        }
        if ($marker) {
            $listAttachRolePoliciesArguments['Marker'] = $marker;
        }
        if ($maxItems) {
            $listAttachRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->iamClient-
>listAttachedRolePolicies($listAttachRolePoliciesArguments);
    }
```

- Pour plus de détails sur l'API, voir ListAttachedRolePoliciesla section Référence des AWS SDK pour PHP API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cette commande renvoie les noms et les ARN des politiques gérées associées au rôle IAM nommé **SecurityAuditRole** dans le compte AWS . Pour consulter la liste des politiques en ligne intégrées au rôle, utilisez la commande **Get-IAMRolePolicyList**.

```
Get-IAMAttachedRolePolicyList -RoleName "SecurityAuditRole"
```

Sortie :

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedRolePoliciesà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cette commande renvoie les noms et les ARN des politiques gérées associées au rôle IAM nommé **SecurityAuditRole** dans le compte AWS . Pour consulter la liste des politiques en ligne intégrées au rôle, utilisez la commande **Get-IAMRolePolicyList**.

```
Get-IAMAttachedRolePolicyList -RoleName "SecurityAuditRole"
```

Sortie :

```
PolicyArn                                                PolicyName
---------                                                ----------
arn:aws:iam::aws:policy/SecurityAudit                    SecurityAudit
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedRolePoliciesà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_attached_policies(role_name):
    """
    Lists policies attached to a role.

    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
        for policy in role.attached_policies.all():
            logger.info("Got policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't list attached policies for %s.", role_name)
        raise
```

- Pour plus de détails sur l'API, consultez ListAttachedRolePoliciesle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, attache et détache les politiques de rôle.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
```

```
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
```

```
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, voir ListAttachedRolePoliciesla section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn list_attached_role_policies(
    client: &iamClient,
    role_name: String,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListAttachedRolePoliciesOutput,
 SdkError<ListAttachedRolePoliciesError>> {
    let response = client
        .list_attached_role_policies()
        .role_name(role_name)
```

```
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Pour plus de détails sur l'API, voir ListAttachedRolePoliciesla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->listattachedrolepolicies(
          iv_rolename = iv_role_name ).
        MESSAGE 'Retrieved attached policy list for role.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role does not exist.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, consultez ListAttachedRolePoliciesla section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3



    /// Returns a list of AWS Identity and Access Management (IAM) policies
    /// that are attached to the role.
    ///
    /// - Parameter role: The IAM role to return the policy list for.
    ///
    /// - Returns: An array of `IAMClientTypes.AttachedPolicy` objects
    ///   describing each managed policy that's attached to the role.
    public func listAttachedRolePolicies(role: String) async throws ->
 [IAMClientTypes.AttachedPolicy] {
        var policyList: [IAMClientTypes.AttachedPolicy] = []

        // Use "Paginated" to get all the attached role polices.
        // This lets the SDK handle the 'isTruncated' in
 "ListAttachedRolePoliciesOutput".
        let input = ListAttachedRolePoliciesInput(
            roleName: role
        )
        let output = client.listAttachedRolePoliciesPaginated(input: input)

        do {
            for try await page in output {
                guard let attachedPolicies = page.attachedPolicies else {
                    print("Error: no attached policies returned.")
                    continue
                }
                for attachedPolicy in attachedPolicies {
                    policyList.append(attachedPolicy)
```

```
                }
            }
        } catch {
            print("ERROR: listAttachedRolePolicies:", dump(error))
            throw error
        }

        return policyList
    }
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedRolePoliciesà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`ListAttachedUserPolicies`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `ListAttachedUserPolicies`.

CLI

AWS CLI

Pour répertorier toutes les politiques gérées attachées à l'utilisateur spécifié

Cette commande renvoie les noms et les ARN des politiques gérées pour l'utilisateur IAM nommé Bob dans le AWS compte.

```
aws iam list-attached-user-policies \
    --user-name Bob
```

Sortie :

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
```

```
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Pour plus d'informations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListAttachedUserPoliciesà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cette commande renvoie les noms et les ARN des politiques gérées pour l'utilisateur IAM nommé **Bob** dans le AWS compte. Pour voir la liste des politiques en ligne qui sont intégrées à l'utilisateur IAM, utilisez la commande **Get-IAMUserPolicyList**.

```
Get-IAMAttachedUserPolicyList -UserName "Bob"
```

Sortie :

```
PolicyArn                                                PolicyName
---------                                                ----------
arn:aws:iam::aws:policy/TesterPolicy                     TesterPolicy
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedUserPoliciesà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cette commande renvoie les noms et les ARN des politiques gérées pour l'utilisateur IAM nommé **Bob** dans le AWS compte. Pour voir la liste des politiques en ligne qui sont intégrées à l'utilisateur IAM, utilisez la commande **Get-IAMUserPolicyList**.

```
Get-IAMAttachedUserPolicyList -UserName "Bob"
```

Sortie :

```
PolicyArn                                                  PolicyName
---------                                                  ----------
arn:aws:iam::aws:policy/TesterPolicy                       TesterPolicy
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedUserPoliciesà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`ListEntitiesForPolicy`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `ListEntitiesForPolicy`.

CLI

AWS CLI

Pour répertorier tous les utilisateurs, groupes et rôles auxquels la politique gérée spécifiée est attachée

Cet exemple renvoie une liste des groupes, des rôles et des utilisateurs IAM auxquels la politique `arn:aws:iam::123456789012:policy/TestPolicy` est attachée.

```
aws iam list-entities-for-policy \
    --policy-arn arn:aws:iam::123456789012:policy/TestPolicy
```

Sortie :

```
{
    "PolicyGroups": [
        {
            "GroupName": "Admins",
            "GroupId": "AGPACKCEVSQ6C2EXAMPLE"
        }
    ],
    "PolicyUsers": [
        {
            "UserName": "Alice",
```

```
            "UserId": "AIDACKCEVSQ6C2EXAMPLE"
        }
    ],
    "PolicyRoles": [
        {
            "RoleName": "DevRole",
            "RoleId": "AROADBQP57FF2AEXAMPLE"
        }
    ],
    "IsTruncated": false
}
```

Pour plus d'informations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListEntitiesForPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie une liste de groupes, de rôles et d'utilisateurs IAM auxquels la politique **`arn:aws:iam::123456789012:policy/TestPolicy`** est attachée.

```
Get-IAMEntitiesForPolicy -PolicyArn "arn:aws:iam::123456789012:policy/TestPolicy"
```

Sortie :

```
IsTruncated  : False
Marker       :
PolicyGroups : {}
PolicyRoles  : {testRole}
PolicyUsers  : {Bob, Theresa}
```

- Pour plus de détails sur l'API, reportez-vous ListEntitiesForPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie une liste de groupes, de rôles et d'utilisateurs IAM auxquels la politique **`arn:aws:iam::123456789012:policy/TestPolicy`** est attachée.

```
Get-IAMEntitiesForPolicy -PolicyArn "arn:aws:iam::123456789012:policy/TestPolicy"
```

Sortie :

```
IsTruncated  : False
Marker       :
PolicyGroups : {}
PolicyRoles  : {testRole}
PolicyUsers  : {Bob, Theresa}
```

- Pour plus de détails sur l'API, reportez-vous ListEntitiesForPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`ListGroupPolicies`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `ListGroupPolicies`.

CLI

AWS CLI

Pour répertorier toutes les politiques en ligne attachées au groupe spécifié

La commande `list-group-policies` suivante répertorie les noms des politiques en ligne attachées au groupe IAM nommé `Admins` dans le compte actuel.

```
aws iam list-group-policies \
    --group-name Admins
```

Sortie :

```
{
    "PolicyNames": [
        "AdminRoot",
        "ExamplePolicy"
```

```
    ]
}
```

Pour plus d'informations, consultez Gestion des politiques IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListGroupPoliciesà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie la liste des politiques en ligne intégrées au groupe **`Testers`**. Pour obtenir les politiques gérées attachées au groupe, utilisez la commande **`Get-IAMAttachedGroupPolicyList`**.

```
Get-IAMGroupPolicyList -GroupName Testers
```

Sortie :

```
Deny-Assume-S3-Role-In-Production
PowerUserAccess-Testers
```

- Pour plus de détails sur l'API, reportez-vous ListGroupPoliciesà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie la liste des politiques en ligne intégrées au groupe **`Testers`**. Pour obtenir les politiques gérées attachées au groupe, utilisez la commande **`Get-IAMAttachedGroupPolicyList`**.

```
Get-IAMGroupPolicyList -GroupName Testers
```

Sortie :

```
Deny-Assume-S3-Role-In-Production
PowerUserAccess-Testers
```

- Pour plus de détails sur l'API, reportez-vous ListGroupPoliciesà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`ListGroups`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `ListGroups`.

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour répertorier les groupes IAM du compte actuel

La commande `list-groups` suivante répertorie les groupes IAM du compte actuel.

```
aws iam list-groups
```

Sortie :

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-06-04T20:27:27.972Z",
            "GroupId": "AIDACKCEVSQ6C2EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "GroupName": "Admins"
        },
        {
            "Path": "/",
            "CreateDate": "2013-04-16T20:30:42Z",
            "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/S3-Admins",
            "GroupName": "S3-Admins"
        }
    ]
}
```

Pour plus d'informations, consultez Gestion des groupes IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// GroupWrapper encapsulates AWS Identity and Access Management (IAM) group
 actions
// used in the examples.
// It contains an IAM service client that is used to perform group actions.
type GroupWrapper struct {
 IamClient *iam.Client
}



// ListGroups lists up to maxGroups number of groups.
func (wrapper GroupWrapper) ListGroups(ctx context.Context, maxGroups int32)
 ([]types.Group, error) {
 var groups []types.Group
 result, err := wrapper.IamClient.ListGroups(ctx, &iam.ListGroupsInput{
  MaxItems: aws.Int32(maxGroups),
 })
 if err != nil {
  log.Printf("Couldn't list groups. Here's why: %v\n", err)
 } else {
  groups = result.Groups
 }
```

```
 return groups, err
}
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section Référence des AWS SDK pour Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les groupes.

```
import { ListGroupsCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 */
export async function* listGroups() {
  const command = new ListGroupsCommand({
    MaxItems: 10,
  });

  let response = await client.send(command);

  while (response.Groups?.length) {
    for (const group of response.Groups) {
      yield group;
    }
```

```
    if (response.IsTruncated) {
      response = await client.send(
        new ListGroupsCommand({
          Marker: response.Marker,
          MaxItems: 10,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listGroups($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listGroupsArguments = [];
        if ($pathPrefix) {
            $listGroupsArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listGroupsArguments["Marker"] = $marker;
        }
        if ($maxItems) {
```

```
            $listGroupsArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listGroups($listGroupsArguments);
    }
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section Référence des AWS SDK pour PHP API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple renvoie une collection de tous les groupes IAM définis dans le fichier actuel Compte AWS.

```
Get-IAMGroupList
```

Sortie :

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE2
GroupName  : Developers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE3
GroupName  : Testers
Path       : /
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple renvoie une collection de tous les groupes IAM définis dans le fichier actuel Compte AWS.

```
Get-IAMGroupList
```

Sortie :

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE2
GroupName  : Developers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE3
GroupName  : Testers
Path       : /
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_groups(count):
```

```
    """
    Lists the specified number of groups for the account.

    :param count: The number of groups to list.
    """
    try:
        for group in iam.groups.limit(count):
            logger.info("Group: %s", group.name)
    except ClientError:
        logger.exception("Couldn't list groups for the account.")
        raise
```

- Pour plus de détails sur l'API, consultez ListGroupsle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# A class to manage IAM operations via the AWS SDK client
class IamGroupManager
  # Initializes the IamGroupManager class
  # @param iam_client [Aws::IAM::Client] An instance of the IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of groups for the account.
  # @param count [Integer] The maximum number of groups to list.
  # @return [Aws::IAM::Client::Response]
  def list_groups(count)
```

```
    response = @iam_client.list_groups(max_items: count)
    response.groups.each do |group|
      @logger.info("\t#{group.group_name}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list groups for the account. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn list_groups(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListGroupsOutput, SdkError<ListGroupsError>> {
    let response = client
        .list_groups()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
```

```
}
```

- Pour plus de détails sur l'API, voir ListGroupsla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->listgroups( ).
        MESSAGE 'Retrieved group list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing groups.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3
```

```
    public func listGroups() async throws -> [String] {
        var groupList: [String] = []

        // Use "Paginated" to get all the groups.
        // This lets the SDK handle the 'isTruncated' property in
"ListGroupsOutput".
        let input = ListGroupsInput()

        let pages = client.listGroupsPaginated(input: input)
        do {
            for try await page in pages {
                guard let groups = page.groups else {
                    print("Error: no groups returned.")
                    continue
                }

                for group in groups {
                    if let name = group.groupName {
                        groupList.append(name)
                    }
                }
            }
        } catch {
            print("ERROR: listGroups:", dump(error))
            throw error
        }
        return groupList
    }
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`ListGroupsForUser`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `ListGroupsForUser`.

CLI

AWS CLI

Pour répertorier les groupes auxquels l'utilisateur IAM appartient

La commande `list-groups-for-user` suivante affiche les groupes auxquels l'utilisateur IAM nommé Bob appartient.

```
aws iam list-groups-for-user \
    --user-name Bob
```

Sortie :

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-05-06T01:18:08Z",
            "GroupId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admin",
            "GroupName": "Admin"
        },
        {
            "Path": "/",
            "CreateDate": "2013-05-06T01:37:28Z",
            "GroupId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/s3-Users",
            "GroupName": "s3-Users"
        }
    ]
}
```

Pour plus d'informations, consultez Gestion des groupes IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListGroupsForUserà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie la liste des groupes IAM auxquels **David** appartient l'utilisateur IAM.

```
Get-IAMGroupForUser -UserName David
```

Sortie :

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE2
GroupName  : Testers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE3
GroupName  : Developers
Path       : /
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsForUserà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie la liste des groupes IAM auxquels **David** appartient l'utilisateur IAM.

```
Get-IAMGroupForUser -UserName David
```

Sortie :

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
```

```
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE2
GroupName  : Testers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE3
GroupName  : Developers
Path       : /
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsForUserà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`ListInstanceProfiles`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `ListInstanceProfiles`.

CLI

AWS CLI

Pour répertorier les profils d'instance du compte

La commande `list-instance-profiles` suivante répertorie les profils d'instance associés au compte actuel.

```
aws iam list-instance-profiles
```

Sortie :

```
{
    "InstanceProfiles": [
```

```
        {
            "Path": "/",
            "InstanceProfileName": "example-dev-role",
            "InstanceProfileId": "AIPAIXEU4NUHUPEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-dev-role",
            "CreateDate": "2023-09-21T18:17:41+00:00",
            "Roles": [
                {
                    "Path": "/",
                    "RoleName": "example-dev-role",
                    "RoleId": "AROAJ52OTH4H7LEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-dev-role",
                    "CreateDate": "2023-09-21T18:17:40+00:00",
                    "AssumeRolePolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
                    }
                }
            ]
        },
        {
            "Path": "/",
            "InstanceProfileName": "example-s3-role",
            "InstanceProfileId": "AIPAJVJVNRIQFREXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-s3-role",
            "CreateDate": "2023-09-21T18:18:50+00:00",
            "Roles": [
                {
                    "Path": "/",
                    "RoleName": "example-s3-role",
                    "RoleId": "AROAINUBC5O7XLEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-s3-role",
                    "CreateDate": "2023-09-21T18:18:49+00:00",
                    "AssumeRolePolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": [
```

```
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
                    }
                }
            ]
        }
    ]
}
```

Pour plus d'informations, consultez Utilisation de profils d'instance dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListInstanceProfilesà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple renvoie une collection des profils d'instance définis dans le fichier actuel Compte AWS.

```
Get-IAMInstanceProfileList
```

Sortie :

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Pour plus de détails sur l'API, reportez-vous ListInstanceProfilesà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple renvoie une collection des profils d'instance définis dans le fichier actuel Compte AWS.

```
Get-IAMInstanceProfileList
```

Sortie :

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Pour plus de détails sur l'API, reportez-vous ListInstanceProfilesà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`ListInstanceProfilesForRole`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `ListInstanceProfilesForRole`.

CLI

AWS CLI

Pour répertorier les profils d'instance d'un rôle IAM

La commande `list-instance-profiles-for-role` suivante répertorie les profils d'instance associés au rôle `Test-Role`.

```
aws iam list-instance-profiles-for-role \
    --role-name Test-Role
```

Sortie :

```
{
    "InstanceProfiles": [
        {
            "InstanceProfileId": "AIDGPMS9RO4H3FEXAMPLE",
            "Roles": [
                {
                    "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
                    "RoleId": "AIDACKCEVSQ6C2EXAMPLE",
                    "CreateDate": "2013-06-07T20:42:15Z",
                    "RoleName": "Test-Role",
                    "Path": "/",
                    "Arn": "arn:aws:iam::123456789012:role/Test-Role"
                }
            ],
            "CreateDate": "2013-06-07T21:05:24Z",
            "InstanceProfileName": "ExampleInstanceProfile",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:instance-profile/
ExampleInstanceProfile"
        }
    ]
}
```

Pour plus d'informations, consultez Utilisation de profils d'instance dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListInstanceProfilesForRoleà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie les détails du profil d'instance associé au rôle **`ec2instancerole`**.

```
Get-IAMInstanceProfileForRole -RoleName ec2instancerole
```

Sortie :

```
      Arn                 : arn:aws:iam::123456789012:instance-profile/
ec2instancerole
```

```
      CreateDate          : 2/17/2015 2:49:04 PM
      InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
      InstanceProfileName : ec2instancerole
      Path                : /
      Roles               : {ec2instancerole}
```

- Pour plus de détails sur l'API, reportez-vous ListInstanceProfilesForRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie les détails du profil d'instance associé au rôle **`ec2instancerole`**.

```
Get-IAMInstanceProfileForRole -RoleName ec2instancerole
```

Sortie :

```
      Arn                 : arn:aws:iam::123456789012:instance-profile/
ec2instancerole
      CreateDate          : 2/17/2015 2:49:04 PM
      InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
      InstanceProfileName : ec2instancerole
      Path                : /
      Roles               : {ec2instancerole}
```

- Pour plus de détails sur l'API, reportez-vous ListInstanceProfilesForRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`ListMfaDevices`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `ListMfaDevices`.

CLI

AWS CLI

Pour répertorier tous les dispositifs MFA d'un utilisateur spécifique

Cet exemple renvoie les informations sur le dispositif MFA attribué à l'utilisateur IAM Bob.

```
aws iam list-mfa-devices \
    --user-name Bob
```

Sortie :

```
{
    "MFADevices": [
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Bob",
            "EnableDate": "2019-10-28T20:37:09+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "GAKT12345678",
            "EnableDate": "2023-02-18T21:44:42+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey1-7XNL7NFNLZ123456789EXAMPLE",
            "EnableDate": "2023-09-19T02:25:35+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey2-VDRQTDBBN5123456789EXAMPLE",
            "EnableDate": "2023-09-19T01:49:18+00:00"
        }
    ]
}
```

Pour plus d'informations, consultez Utilisation de l'authentification multifactorielle (MFA) dans AWS dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListMfaDevicesà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie des informations sur le dispositif MFA attribué à l'utilisateur IAM **David**. Dans cet exemple, vous pouvez voir qu'il s'agit d'un dispositif virtuel, car le **SerialNumber** est un ARN au lieu du numéro de série réel du dispositif physique.

```
Get-IAMMFADevice -UserName David
```

Sortie :

```
EnableDate                SerialNumber                            UserName
----------                ------------                            --------
4/8/2015 9:41:10 AM       arn:aws:iam::123456789012:mfa/David     David
```

- Pour plus de détails sur l'API, reportez-vous ListMfaDevicesà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie des informations sur le dispositif MFA attribué à l'utilisateur IAM **David**. Dans cet exemple, vous pouvez voir qu'il s'agit d'un dispositif virtuel, car le **SerialNumber** est un ARN au lieu du numéro de série réel du dispositif physique.

```
Get-IAMMFADevice -UserName David
```

Sortie :

```
EnableDate                SerialNumber                            UserName
----------                ------------                            --------
4/8/2015 9:41:10 AM       arn:aws:iam::123456789012:mfa/David     David
```

- Pour plus de détails sur l'API, reportez-vous ListMfaDevicesà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`ListOpenIdConnectProviders`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `ListOpenIdConnectProviders`.

CLI

AWS CLI

Pour répertorier les informations relatives aux fournisseurs OpenID Connect dans le compte AWS

Cet exemple renvoie une liste des ARNS de tous les fournisseurs OpenID Connect définis dans AWS le compte courant.

```
aws iam list-open-id-connect-providers
```

Sortie :

```
{
    "OpenIDConnectProviderList": [
        {
            "Arn": "arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com"
        }
    ]
}
```

Pour plus d'informations, consultez Création de fournisseurs d'identité OpenID Connect (OIDC) du Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListOpenIdConnectProvidersà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie une liste des ARN de tous les fournisseurs OpenID Connect définis dans l' Compte AWS actuel.

```
Get-IAMOpenIDConnectProviderList
```

Sortie :

```
Arn
---
arn:aws:iam::123456789012:oidc-provider/server.example.com
arn:aws:iam::123456789012:oidc-provider/another.provider.com
```

- Pour plus de détails sur l'API, reportez-vous ListOpenIdConnectProvidersà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie une liste des ARN de tous les fournisseurs OpenID Connect définis dans l' Compte AWS actuel.

```
Get-IAMOpenIDConnectProviderList
```

Sortie :

```
Arn
---
arn:aws:iam::123456789012:oidc-provider/server.example.com
arn:aws:iam::123456789012:oidc-provider/another.provider.com
```

- Pour plus de détails sur l'API, reportez-vous ListOpenIdConnectProvidersà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`ListPolicies`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `ListPolicies`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer les politiques

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::listPolicies(const Aws::Client::ClientConfiguration
 &clientConfig) {
    const Aws::String DATE_FORMAT("%Y-%m-%d");
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListPoliciesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListPolicies(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam policies: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
                      std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                      std::setw(64) << "Description" << std::setw(12) <<
                      "CreateDate" << std::endl;
            header = true;
        }

        const auto &policies = outcome.GetResult().GetPolicies();
        for (const auto &policy: policies) {
            std::cout << std::left << std::setw(55) <<
                      policy.GetPolicyName() << std::setw(30) <<
                      policy.GetPolicyId() << std::setw(80) << policy.GetArn() <<
                      std::setw(64) << policy.GetDescription() << std::setw(12)
 <<
                      policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
```

```
                        std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour répertorier les politiques gérées disponibles pour votre AWS compte

Cet exemple renvoie une collection des deux premières politiques gérées disponibles dans le AWS compte courant.

```
aws iam list-policies \
    --max-items 3
```

Sortie :

```
{
    "Policies": [
        {
            "PolicyName": "AWSCloudTrailAccessPolicy",
            "PolicyId": "ANPAXQE2B5PJ7YEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:policy/AWSCloudTrailAccessPolicy",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 0,
```

```
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2019-09-04T17:43:42+00:00",
            "UpdateDate": "2019-09-04T17:43:42+00:00"
        },
        {
            "PolicyName": "AdministratorAccess",
            "PolicyId": "ANPAIWMBCKSKIEE64ZLYK",
            "Arn": "arn:aws:iam::aws:policy/AdministratorAccess",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 6,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:46+00:00",
            "UpdateDate": "2015-02-06T18:39:46+00:00"
        },
        {
            "PolicyName": "PowerUserAccess",
            "PolicyId": "ANPAJYRXTHIB4FOVS3ZXS",
            "Arn": "arn:aws:iam::aws:policy/PowerUserAccess",
            "Path": "/",
            "DefaultVersionId": "v5",
            "AttachmentCount": 1,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:47+00:00",
            "UpdateDate": "2023-07-06T22:04:00+00:00"
        }
    ],
    "NextToken": "EXAMPLErZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiA4fQ=="
}
```


Pour plus d'informations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(ctx context.Context, maxPolicies int32)
 ([]types.Policy, error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
```

```
 }
 return policies, err
}
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK pour Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les politiques.

```
import { ListPoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listPolicies() {
  const command = new ListPoliciesCommand({
    MaxItems: 10,
    OnlyAttached: false,
    // List only the customer managed policies in your Amazon Web Services
 account.
    Scope: "Local",
  });
```

```
  let response = await client.send(command);

  while (response.Policies?.length) {
    for (const policy of response.Policies) {
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListPoliciesCommand({
          Marker: response.Marker,
          MaxItems: 10,
          OnlyAttached: false,
          Scope: "Local",
        }),
      );
    } else {
      break;
    }
  }
}
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listPolicies($pathPrefix = "", $marker = "", $maxItems = 0)
    {
```

```
        $listPoliciesArguments = [];
        if ($pathPrefix) {
            $listPoliciesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listPoliciesArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listPoliciesArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listPolicies($listPoliciesArguments);
    }
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK pour PHP API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple renvoie une collection des trois premières politiques gérées disponibles dans le AWS compte courant. Comme il n'**-scope**est pas spécifié, il utilise par défaut **all** et inclut à la fois des politiques AWS gérées et des politiques gérées par le client.

```
Get-IAMPolicyList -MaxItem 3
```

Sortie :

```
Arn              : arn:aws:iam::aws:policy/AWSDirectConnectReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : AWSDirectConnectReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:08 AM
```

```
Arn              : arn:aws:iam::aws:policy/AmazonGlacierReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:27 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : NJKMU274MET4EEXAMPLE2
PolicyName       : AmazonGlacierReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:27 AM

Arn              : arn:aws:iam::aws:policy/AWSMarketplaceFullAccess
AttachmentCount  : 0
CreateDate       : 2/11/2015 9:21:45 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : 5ULJSO2FYVPYGEXAMPLE3
PolicyName       : AWSMarketplaceFullAccess
UpdateDate       : 2/11/2015 9:21:45 AM
```

Exemple 2 : Cet exemple renvoie un ensemble des deux premières politiques gérées par le client disponibles dans le AWS compte courant. Il utilise **`-Scope local`** pour limiter la sortie aux seules politiques gérées par le client.

```
Get-IAMPolicyList -Scope local -MaxItem 2
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:policy/MyLocalPolicy
AttachmentCount  : 0
CreateDate       : 2/12/2015 9:39:09 AM
DefaultVersionId : v2
Description      :
IsAttachable     : True
Path             : /
PolicyId         : SQVCBLC4VAOUCEXAMPLE4
PolicyName       : MyLocalPolicy
UpdateDate       : 2/12/2015 9:39:53 AM

Arn              : arn:aws:iam::123456789012:policy/policyforec2instancerole
```

```
AttachmentCount  : 1
CreateDate       : 2/17/2015 2:51:38 PM
DefaultVersionId : v11
Description      :
IsAttachable     : True
Path             : /
PolicyId         : X5JPBLJH2Z2SOEXAMPLE5
PolicyName       : policyforec2instancerole
UpdateDate       : 2/18/2015 8:52:31 AM
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple renvoie une collection des trois premières politiques gérées disponibles dans le AWS compte courant. Comme il n'**`-scope`**est pas spécifié, il utilise par défaut **`all`** et inclut à la fois des politiques AWS gérées et des politiques gérées par le client.

```
Get-IAMPolicyList -MaxItem 3
```

Sortie :

```
Arn              : arn:aws:iam::aws:policy/AWSDirectConnectReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : AWSDirectConnectReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:08 AM

Arn              : arn:aws:iam::aws:policy/AmazonGlacierReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:27 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : NJKMU274MET4EEXAMPLE2
PolicyName       : AmazonGlacierReadOnlyAccess
```

```
UpdateDate       : 2/6/2015 10:40:27 AM

Arn              : arn:aws:iam::aws:policy/AWSMarketplaceFullAccess
AttachmentCount  : 0
CreateDate       : 2/11/2015 9:21:45 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : 5ULJSO2FYVPYGEXAMPLE3
PolicyName       : AWSMarketplaceFullAccess
UpdateDate       : 2/11/2015 9:21:45 AM
```

Exemple 2 : Cet exemple renvoie un ensemble des deux premières politiques gérées par le client disponibles dans le AWS compte courant. Il utilise **`-Scope local`** pour limiter la sortie aux seules politiques gérées par le client.

```
Get-IAMPolicyList -Scope local -MaxItem 2
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:policy/MyLocalPolicy
AttachmentCount  : 0
CreateDate       : 2/12/2015 9:39:09 AM
DefaultVersionId : v2
Description      :
IsAttachable     : True
Path             : /
PolicyId         : SQVCBLC4VAOUCEXAMPLE4
PolicyName       : MyLocalPolicy
UpdateDate       : 2/12/2015 9:39:53 AM

Arn              : arn:aws:iam::123456789012:policy/policyforec2instancerole
AttachmentCount  : 1
CreateDate       : 2/17/2015 2:51:38 PM
DefaultVersionId : v11
Description      :
IsAttachable     : True
Path             : /
PolicyId         : X5JPBLJH2Z2SOEXAMPLE5
PolicyName       : policyforec2instancerole
UpdateDate       : 2/18/2015 8:52:31 AM
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_policies(scope):
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
                  'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
        return policies
```

- Pour plus de détails sur l'API, consultez ListPolicieslle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, attache et détache les politiques de rôle.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
```

```
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
```

```
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn list_policies(
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
```

```
        .try_collect()
        .await?;

    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- Pour plus de détails sur l'API, voir ListPolicieslа section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->listpolicies( iv_scope = iv_scope ).
        MESSAGE 'Retrieved policy list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing policies.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous ListPolicieslà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func listPolicies() async throws -> [MyPolicyRecord] {
        var policyList: [MyPolicyRecord] = []

        // Use "Paginated" to get all the policies.
        // This lets the SDK handle the 'isTruncated' in "ListPoliciesOutput".
        let input = ListPoliciesInput()
        let output = client.listPoliciesPaginated(input: input)

        do {
            for try await page in output {
                guard let policies = page.policies else {
                    print("Error: no policies returned.")
                    continue
                }

                for policy in policies {
                    guard let name = policy.policyName,
                          let id = policy.policyId,
                          let arn = policy.arn
                    else {
                        throw ServiceHandlerError.noSuchPolicy
                    }
                    policyList.append(MyPolicyRecord(name: name, id: id, arn:
 arn))
                }
            }
        } catch {
            print("ERROR: listPolicies:", dump(error))
```

```
            throw error
        }

        return policyList
    }
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`ListPolicyVersions`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `ListPolicyVersions`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Gérer les politiques
- Restaurer une version de stratégie

CLI

AWS CLI

Pour récupérer des informations sur les versions de la politique gérée spécifiée

Cet exemple renvoie la liste des versions disponibles de la politique dont l'ARN est `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam list-policy-versions \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Sortie :

```
{
```

```
    "IsTruncated": false,
    "Versions": [
        {
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2015-06-02T23:19:44Z"
        },
        {
        "VersionId": "v1",
        "IsDefaultVersion": false,
        "CreateDate": "2015-06-02T22:30:47Z"
        }
    ]
}
```

Pour plus d'informations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir ListPolicyVersionsla section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie la liste des versions disponibles de la politique dont l'ARN est **`arn:aws:iam::123456789012:policy/MyManagedPolicy`**. Pour obtenir le document de politique pour une version spécifique, utilisez la commande **`Get-IAMPolicyVersion`** et spécifiez le **`VersionId`** de celle que vous souhaitez.

```
Get-IAMPolicyVersionList -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy
```

Sortie :

```
CreateDate                  Document                IsDefaultVersion
       VersionId
----------                  --------                ----------------
       ---------
2/12/2015 9:39:53 AM                                True
       v2
```

```
2/12/2015 9:39:09 AM                                    False
       v1
```

- Pour plus de détails sur l'API, reportez-vous ListPolicyVersionsà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie la liste des versions disponibles de la politique dont l'ARN est **`arn:aws:iam::123456789012:policy/MyManagedPolicy`**. Pour obtenir le document de politique pour une version spécifique, utilisez la commande **`Get-IAMPolicyVersion`** et spécifiez le **`VersionId`** de celle que vous souhaitez.

```
Get-IAMPolicyVersionList -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy
```

Sortie :

```
CreateDate                  Document                IsDefaultVersion
       VersionId
----------                  --------                ----------------
       ---------
2/12/2015 9:39:53 AM                                True
       v2
2/12/2015 9:39:09 AM                                False
       v1
```

- Pour plus de détails sur l'API, reportez-vous ListPolicyVersionsà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->listpolicyversions(
          iv_policyarn = iv_policy_arn ).
        MESSAGE 'Retrieved policy versions list.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing policy versions.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, consultez ListPolicyVersionsla section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`ListRolePolicies`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `ListRolePolicies`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Débuter avec la protection des appareils liés à l'Internet des objets

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
```

```
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }
```

- Pour plus de détails sur l'API, reportez-vous ListRolePoliciesà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour répertorier les politiques attachées à un rôle IAM

La commande `list-role-policies` suivante répertorie les noms des politiques des autorisations pour le rôle IAM spécifié.

```
aws iam list-role-policies \
    --role-name Test-Role
```

Sortie :

```
{
    "PolicyNames": [
```

```
        "ExamplePolicy"
    ]
}
```

Pour voir la politique d'approbation attachée à un rôle, utilisez la commande `get-role`. Pour voir les détails d'une politique des autorisations, utilisez la commande `get-role-policy`.

Pour plus d'informations, consultez Création de rôles IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListRolePoliciesà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}
```

```
// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(ctx context.Context, roleName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(ctx,
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

- Pour plus de détails sur l'API, reportez-vous ListRolePoliciesà la section Référence des AWS SDK pour Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les politiques.

```
import { ListRolePoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
```

```
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} roleName
 */
export async function* listRolePolicies(roleName) {
  const command = new ListRolePoliciesCommand({
    RoleName: roleName,
    MaxItems: 10,
  });

  let response = await client.send(command);

  while (response.PolicyNames?.length) {
    for (const policyName of response.PolicyNames) {
      yield policyName;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolePoliciesCommand({
          RoleName: roleName,
          MaxItems: 10,
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Pour plus de détails sur l'API, reportez-vous ListRolePoliciesà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listRolePolicies($roleName, $marker = "", $maxItems = 0)
    {
        $listRolePoliciesArguments = ['RoleName' => $roleName];
        if ($marker) {
            $listRolePoliciesArguments['Marker'] = $marker;
        }
        if ($maxItems) {
            $listRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->customWaiter(function () use ($listRolePoliciesArguments) {
            return $this->iamClient-
>listRolePolicies($listRolePoliciesArguments);
        });
    }
```

- Pour plus de détails sur l'API, reportez-vous ListRolePoliciesà la section Référence des AWS SDK pour PHP API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renvoie la liste des noms des politiques en ligne intégrées au rôle IAM **`lamda_exec_role`**. Pour consulter les détails d'une politique en ligne, utilisez la commande **`Get-IAMRolePolicy`**.

```
Get-IAMRolePolicyList -RoleName lambda_exec_role
```

Sortie :

```
oneClick_lambda_exec_role_policy
```

- Pour plus de détails sur l'API, reportez-vous ListRolePoliciesà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renvoie la liste des noms des politiques en ligne intégrées au rôle IAM **`lamda_exec_role`**. Pour consulter les détails d'une politique en ligne, utilisez la commande **`Get-IAMRolePolicy`**.

```
Get-IAMRolePolicyList -RoleName lambda_exec_role
```

Sortie :

```
oneClick_lambda_exec_role_policy
```

- Pour plus de détails sur l'API, reportez-vous ListRolePoliciesà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_policies(role_name):
    """
    Lists inline policies for a role.
```

```
    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
        for policy in role.policies.all():
            logger.info("Got inline policy %s.", policy.name)
    except ClientError:
        logger.exception("Couldn't list inline policies for %s.", role_name)
        raise
```

- Pour plus de détails sur l'API, consultez ListRolePolicieslе AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# Lists policy ARNs attached to a role
#
# @param role_name [String] The name of the role
# @return [Array<String>] List of policy ARNs
def list_attached_policy_arns(role_name)
  response = @iam_client.list_attached_role_policies(role_name: role_name)
  response.attached_policies.map(&:policy_arn)
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing policies attached to role: #{e.message}")
  []
end
```

- Pour plus de détails sur l'API, reportez-vous ListRolePoliciesà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn list_role_policies(
    client: &iamClient,
    role_name: &str,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolePoliciesOutput, SdkError<ListRolePoliciesError>> {
    let response = client
        .list_role_policies()
        .role_name(role_name)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Pour plus de détails sur l'API, voir ListRolePoliciesla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->listrolepolicies(
          iv_rolename = iv_role_name ).
        MESSAGE 'Retrieved inline policy list for role.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role does not exist.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous ListRolePoliciesà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func listRolePolicies(role: String) async throws -> [String] {
        var policyList: [String] = []

        // Use "Paginated" to get all the role policies.
        // This lets the SDK handle the 'isTruncated' in
 "ListRolePoliciesOutput".
        let input = ListRolePoliciesInput(
            roleName: role
        )
        let pages = client.listRolePoliciesPaginated(input: input)

        do {
            for try await page in pages {
                guard let policies = page.policyNames else {
                    print("Error: no role policies returned.")
```

```
                    continue
                }

                for policy in policies {
                    policyList.append(policy)
                }
            }
        } catch {
            print("ERROR: listRolePolicies:", dump(error))
            throw error
        }
        return policyList
    }
```

- Pour plus de détails sur l'API, reportez-vous ListRolePoliciesà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`ListRoleTags`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `ListRoleTags`.

CLI

AWS CLI

Pour répertorier les balises attachées à un rôle

La commande `list-role-tags` suivante extrait la liste des balises associées au rôle spécifié.

```
aws iam list-role-tags \
    --role-name production-role
```

Sortie :

```
{
    "Tags": [
```

```
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

Pour plus d'informations, consultez Étiquetage de ressources IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListRoleTagsà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple récupère la balise associée au rôle.

```
Get-IAMRoleTagList -RoleName MyRoleName
```

- Pour plus de détails sur l'API, reportez-vous ListRoleTagsà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple récupère la balise associée au rôle.

```
Get-IAMRoleTagList -RoleName MyRoleName
```

- Pour plus de détails sur l'API, reportez-vous ListRoleTagsà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`ListRoles`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `ListRoles`.

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour répertorier les rôles IAM du compte actuel

La commande `list-roles` suivante répertorie les rôles IAM du compte actuel.

```
aws iam list-roles
```

Sortie :

```
{
    "Roles": [
        {
            "Path": "/",
            "RoleName": "ExampleRole",
            "RoleId": "AROAJ52OTH4H7LEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/ExampleRole",
            "CreateDate": "2017-09-12T19:23:36+00:00",
            "AssumeRolePolicyDocument": {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        },
        {
            "Path": "/example_path/",
            "RoleName": "ExampleRoleWithPath",
            "RoleId": "AROAI4QRP7UFT7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/example_path/
ExampleRoleWithPath",
            "CreateDate": "2023-09-21T20:29:38+00:00",
            "AssumeRolePolicyDocument": {
```

```
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        }
    ]
}
```

Pour plus d'informations, consultez Création de rôles IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
```

```
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(ctx context.Context, maxRoles int32)
 ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(ctx,
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section Référence des AWS SDK pour Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les rôles.

```
import { ListRolesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listRoles() {
  const command = new ListRolesCommand({
    MaxItems: 10,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListRolesCommandOutput | undefined}
   */
  let response = await client.send(command);

  while (response?.Roles?.length) {
    for (const role of response.Roles) {
      yield role;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolesCommand({
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    /**
     * @param string $pathPrefix
     * @param string $marker
     * @param int $maxItems
     * @return Result
     * $roles = $service->listRoles();
     */
    public function listRoles($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listRolesArguments = [];
        if ($pathPrefix) {
            $listRolesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listRolesArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listRolesArguments["MaxItems"] = $maxItems;
        }
        return $this->iamClient->listRoles($listRolesArguments);
    }
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section Référence des AWS SDK pour PHP API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple extrait une liste de tous les rôles IAM dans l Compte AWS.

```
Get-IAMRoleList
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple extrait une liste de tous les rôles IAM dans l Compte AWS.

```
Get-IAMRoleList
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_roles(count):
    """
    Lists the specified number of roles for the account.

    :param count: The number of roles to list.
    """
    try:
        roles = list(iam.roles.limit(count=count))
        for role in roles:
            logger.info("Role: %s", role.name)
    except ClientError:
```

```
        logger.exception("Couldn't list roles for the account.")
        raise
    else:
        return roles
```

- Pour plus de détails sur l'API, consultez ListRolesle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# Lists IAM roles up to a specified count.
# @param count [Integer] the maximum number of roles to list.
# @return [Array<String>] the names of the roles.
def list_roles(count)
  role_names = []
  roles_counted = 0

  @iam_client.list_roles.each_page do |page|
    page.roles.each do |role|
      break if roles_counted >= count

      @logger.info("\t#{roles_counted + 1}: #{role.role_name}")
      role_names << role.role_name
      roles_counted += 1
    end
    break if roles_counted >= count
  end

  role_names
rescue Aws::IAM::Errors::ServiceError => e
```

```
    @logger.error("Couldn't list roles for the account. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn list_roles(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolesOutput, SdkError<ListRolesError>> {
    let response = client
        .list_roles()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;
    Ok(response)
}
```

- Pour plus de détails sur l'API, voir ListRolesla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->listroles( ).
        MESSAGE 'Retrieved role list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing roles.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func listRoles() async throws -> [String] {
        var roleList: [String] = []

        // Use "Paginated" to get all the roles.
```

```
        // This lets the SDK handle the 'isTruncated' in "ListRolesOutput".
        let input = ListRolesInput()
        let pages = client.listRolesPaginated(input: input)

        do {
            for try await page in pages {
                guard let roles = page.roles else {
                    print("Error: no roles returned.")
                    continue
                }

                for role in roles {
                    if let name = role.roleName {
                        roleList.append(name)
                    }
                }
            }
        } catch {
            print("ERROR: listRoles:", dump(error))
            throw error
        }
        return roleList
    }
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisez **`ListSamlProviders`** avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `ListSAMLProviders`.

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }
```

- Pour les détails de l'API, consultez ListSAMLProviders dans la Référence de l'API AWS SDK pour .NET .

CLI

AWS CLI

Pour répertorier les fournisseurs SAML dans le compte AWS

Cet exemple extrait la liste des fournisseurs SAML 2.0 créés dans le compte courant AWS .

```
aws iam list-saml-providers
```

Sortie :

```
{
    "SAMLProviderList": [
```

```
        {
            "Arn": "arn:aws:iam::123456789012:saml-provider/SAML-ADFS",
            "ValidUntil": "2015-06-05T22:45:14Z",
            "CreateDate": "2015-06-05T22:45:14Z"
        }
    ]
}
```

Pour plus d'informations, consultez Création de fournisseurs d'identité SAML IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus d'informations sur l'API, consultez ListSAMLProviders dans la Référence des commandes AWS CLI .

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}
```

```
// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders(ctx context.Context)
 ([]types.SAMLProviderListEntry, error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(ctx,
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

- Pour les détails de l'API, consultez ListSAMLProviders dans la Référence de l'API AWS SDK pour Go .

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les fournisseurs SAML.

```
import { ListSAMLProvidersCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listSamlProviders = async () => {
  const command = new ListSAMLProvidersCommand({});

  const response = await client.send(command);
```

```
  console.log(response);
  return response;
};
```

- Pour les détails de l'API, consultez ListSAMLProviders dans la Référence de l'API AWS SDK pour JavaScript .

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listSAMLProviders()
    {
        return $this->iamClient->listSAMLProviders();
    }
```

- Pour les détails de l'API, consultez ListSAMLProviders dans la Référence de l'API AWS SDK pour PHP .

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple extrait la liste des fournisseurs SAML 2.0 créés dans l' Compte AWS actuel. Il renvoie l'ARN, la date de création et la date d'expiration pour chaque fournisseur SAML.

```
Get-IAMSAMLProviderList
```

Sortie :

```
Arn                                                 CreateDate
   ValidUntil
---                                                 ----------
   ----------
arn:aws:iam::123456789012:saml-provider/SAMLADFS    12/23/2014 12:16:55 PM
   12/23/2114 12:16:54 PM
```

- Pour plus de détails sur l'API, consultez ListSAMLProviders dans la Référence des applets de commande pour les Outils AWS pour PowerShell (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple extrait la liste des fournisseurs SAML 2.0 créés dans l' Compte AWS actuel. Il renvoie l'ARN, la date de création et la date d'expiration pour chaque fournisseur SAML.

```
Get-IAMSAMLProviderList
```

Sortie :

```
Arn                                                 CreateDate
   ValidUntil
---                                                 ----------
   ----------
arn:aws:iam::123456789012:saml-provider/SAMLADFS    12/23/2014 12:16:55 PM
   12/23/2114 12:16:54 PM
```

- Pour plus de détails sur l'API, consultez ListSAMLProviders dans la Référence des applets de commande Outils AWS pour PowerShell (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_saml_providers(count):
    """
    Lists the SAML providers for the account.

    :param count: The maximum number of providers to list.
    """
    try:
        found = 0
        for provider in iam.saml_providers.limit(count):
            logger.info("Got SAML provider %s.", provider.arn)
            found += 1
        if found == 0:
            logger.info("Your account has no SAML providers.")
    except ClientError:
        logger.exception("Couldn't list SAML providers.")
        raise
```

- Pour les détails de l'API, consultez ListSAMLProviders dans la Référence de l'API du kit SDK AWS pour Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class SamlProviderLister
  # Initializes the SamlProviderLister with IAM client and a logger.
  # @param iam_client [Aws::IAM::Client] The IAM client object.
  # @param logger [Logger] The logger object for logging output.
  def initialize(iam_client, logger = Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end
```

```
  # Lists up to a specified number of SAML providers for the account.
  # @param count [Integer] The maximum number of providers to list.
  # @return [Aws::IAM::Client::Response]
  def list_saml_providers(count)
    response = @iam_client.list_saml_providers
    response.saml_provider_list.take(count).each do |provider|
      @logger.info("\t#{provider.arn}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list SAML providers. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Pour les détails de l'API, consultez ListSAMLProviders dans la Référence de l'API AWS SDK pour Ruby .

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn list_saml_providers(
    client: &Client,
) -> Result<ListSamlProvidersOutput, SdkError<ListSAMLProvidersError>> {
    let response = client.list_saml_providers().send().await?;

    Ok(response)
}
```

- Pour les détails de l'API, consultez ListSAMLProviders dans la Référence de l'API du kit SDK AWS pour Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    oo_result = lo_iam->listsamlproviders( ).
    MESSAGE 'Retrieved SAML provider list.' TYPE 'I'.
  CATCH /aws1/cx_iamservicefailureex.
    MESSAGE 'Service failure when listing SAML providers.' TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, consultez ListSamlProviders dans le AWS SDK pour la référence de l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`ListServerCertificates`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `ListServerCertificates`.

C++

Kit de développement logiciel (SDK) for C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::listServerCertificates(
        const Aws::Client::ClientConfiguration &clientConfig) {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";

    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListServerCertificatesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListServerCertificates(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list server certificates: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
                      std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                      std::setw(14) << "UploadDate" << std::setw(14) <<
                      "ExpirationDate" << std::endl;
            header = true;
        }

        const auto &certificates =
                outcome.GetResult().GetServerCertificateMetadataList();

        for (const auto &certificate: certificates) {
            std::cout << std::left << std::setw(55) <<
                      certificate.GetServerCertificateName() << std::setw(30) <<
                      certificate.GetServerCertificateId() << std::setw(80) <<
```

```
                    certificate.GetArn() << std::setw(14) <<

 certificate.GetUploadDate().ToGmtString(DATE_FORMAT.c_str()) <<
                    std::setw(14) <<

 certificate.GetExpiration().ToGmtString(DATE_FORMAT.c_str()) <<
                    std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Pour plus de détails sur l'API, reportez-vous ListServerCertificatesà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour répertorier les certificats de serveur de votre AWS compte

La `list-server-certificates` commande suivante répertorie tous les certificats de serveur stockés et disponibles pour utilisation dans votre AWS compte.

```
aws iam list-server-certificates
```

Sortie :

```
{
    "ServerCertificateMetadataList": [
        {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
```

```
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        {
            "Path": "/cloudfront/",
            "ServerCertificateName": "MyTestCert",
            "ServerCertificateId": "ASCAEXAMPLE456EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyTestCert",
            "UploadDate": "2015-04-21T18:14:16+00:00",
            "Expiration": "2018-01-14T17:52:36+00:00"
        }
    ]
}
```

Pour plus d'informations, consultez Gestion des certificats de serveur dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListServerCertificatesà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les certificats.

```
import { ListServerCertificatesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
```

```
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listServerCertificates() {
  const command = new ListServerCertificatesCommand({});
  let response = await client.send(command);

  while (response.ServerCertificateMetadataList?.length) {
    for await (const cert of response.ServerCertificateMetadataList) {
      yield cert;
    }

    if (response.IsTruncated) {
      response = await client.send(new ListServerCertificatesCommand({}));
    } else {
      break;
    }
  }
}
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous ListServerCertificatesà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });
```

```
iam.listServerCertificates({}, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous ListServerCertificatesà la section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple extrait la liste des certificats de serveur qui ont été chargés sur l'Compte AWS actuel.

```
Get-IAMServerCertificateList
```

Sortie :

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Pour plus de détails sur l'API, reportez-vous ListServerCertificatesà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple extrait la liste des certificats de serveur qui ont été chargés sur l'Compte AWS actuel.

```
Get-IAMServerCertificateList
```

Sortie :

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Pour plus de détails sur l'API, reportez-vous ListServerCertificatesà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez, mettez à jour et supprimez des certificats de serveur.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'ServerCertificateManager'
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
```

```
                                          server_certificate_name: name,
                                          certificate_body: certificate_body,
                                          private_key: private_key
                                        })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info('No server certificates found.')
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
#{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
'#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
```

```
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, reportez-vous ListServerCertificatesà la section Référence des AWS SDK pour Ruby API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`ListSigningCertificates`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `ListSigningCertificates`.

CLI

AWS CLI

Pour répertorier les certificats de signature d'un utilisateur IAM

La commande `list-signing-certificates` suivante répertorie les certificats de signature de l'utilisateur IAM nommé Bob.

```
aws iam list-signing-certificates \
    --user-name Bob
```

Sortie :

```
{
    "Certificates": [
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-
body>-----END CERTIFICATE-----",
            "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
```

```
            "UploadDate": "2013-06-06T21:40:08Z"
        }
    ]
}
```

Pour plus d'informations, consultez Gestion des certificats de signature dans le Guide de l'utilisateur Amazon EC2.

- Pour plus de détails sur l'API, reportez-vous ListSigningCertificatesà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple extrait les détails du certificat de signature associé à l'utilisateur nommé **Bob**.

```
Get-IAMSigningCertificate -UserName Bob
```

Sortie :

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                    YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                    21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
```

```
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Pour plus de détails sur l'API, reportez-vous ListSigningCertificatesà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple extrait les détails du certificat de signature associé à l'utilisateur nommé **Bob**.

```
Get-IAMSigningCertificate -UserName Bob
```

Sortie :

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
```

```
                    YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                    21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                    rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                    nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                    NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                    -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Pour plus de détails sur l'API, reportez-vous ListSigningCertificatesà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`ListUserPolicies`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `ListUserPolicies`.

CLI

AWS CLI

Pour répertorier les politiques d'un utilisateur IAM

La commande `list-user-policies` suivante répertorie les politiques attachées à l'utilisateur IAM nommé Bob.

```
aws iam list-user-policies \
    --user-name Bob
```

Sortie :

```
{
    "PolicyNames": [
        "ExamplePolicy",
        "TestPolicy"
    ]
}
```

Pour plus d'informations, consultez la section Création d'un utilisateur IAM dans votre AWS compte dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListUserPoliciesà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
```

```
 IamClient *iam.Client
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(ctx context.Context, userName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(ctx,
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

- Pour plus de détails sur l'API, reportez-vous ListUserPoliciesà la section Référence des AWS SDK pour Go API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple extrait la liste des noms des politiques en ligne intégrées à l'utilisateur IAM nommé **`David`**.

```
Get-IAMUserPolicyList -UserName David
```

Sortie :

```
Davids_IAM_Admin_Policy
```

- Pour plus de détails sur l'API, reportez-vous ListUserPoliciesà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple extrait la liste des noms des politiques en ligne intégrées à l'utilisateur IAM nommé **David**.

```
Get-IAMUserPolicyList -UserName David
```

Sortie :

```
Davids_IAM_Admin_Policy
```

- Pour plus de détails sur l'API, reportez-vous ListUserPoliciesà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **ListUserTags** avec une CLI

Les exemples de code suivants illustrent comment utiliser `ListUserTags`.

CLI

AWS CLI

Pour répertorier les balises attachées à un utilisateur

La commande `list-user-tags` suivante extrait la liste des balises associées à l'utilisateur IAM spécifié.

```
aws iam list-user-tags \
    --user-name alice
```

Sortie :

```
{
    "Tags": [
```

```
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

Pour plus d'informations, consultez Étiquetage de ressources IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListUserTagsà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple extrait la balise associée à l'utilisateur.

```
Get-IAMUserTagList -UserName joe
```

- Pour plus de détails sur l'API, reportez-vous ListUserTagsà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple extrait la balise associée à l'utilisateur.

```
Get-IAMUserTagList -UserName joe
```

- Pour plus de détails sur l'API, reportez-vous ListUserTagsà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`ListUsers`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `ListUsers`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Créer des utilisateurs en lecture seule et en lecture-écriture

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK pour .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_list_users
#
# List the IAM users in the account.
#
# Returns:
#       The list of users names
#    And:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_list_users() {
  local option OPTARG # Required to use getopts command in a function.
  local error_code
  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_users"
    echo "Lists the AWS Identity and Access Management (IAM) user in the
 account."
```

```
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "h" option; do
    case "${option}" in
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local response

  response=$(aws iam list-users \
    --output text \
    --query "Users[].UserName")
  error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-users operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::listUsers(const Aws::Client::ClientConfiguration &clientConfig)
 {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListUsersRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListUsers(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam users:" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(32) << "Name" <<
                      std::setw(30) << "ID" << std::setw(64) << "Arn" <<
                      std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &users = outcome.GetResult().GetUsers();
        for (const auto &user: users) {
            std::cout << std::left << std::setw(32) << user.GetUserName() <<
                      std::setw(30) << user.GetUserId() << std::setw(64) <<
                      user.GetArn() << std::setw(20) <<
                      user.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
                      << std::endl;
        }
```

```
        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour répertorier les utilisateurs IAM

La commande `list-users` suivante répertorie les utilisateurs IAM du compte actuel.

```
aws iam list-users
```

Sortie :

```
{
    "Users": [
        {
            "UserName": "Adele",
            "Path": "/",
            "CreateDate": "2013-03-07T05:14:48Z",
            "UserId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Adele"
        },
        {
            "UserName": "Bob",
            "Path": "/",
            "CreateDate": "2012-09-21T23:03:13Z",
            "UserId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Bob"
```

```
        }
    ]
}
```

Pour plus d'informations, consultez Liste des utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
```

```
func (wrapper UserWrapper) ListUsers(ctx context.Context, maxUsers int32)
 ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(ctx, &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK pour Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.AttachedPermissionsBoundary;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListUsersRequest;
import software.amazon.awssdk.services.iam.model.ListUsersResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.User;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
```

```
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListUsers {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listAllUsers(iam);
        System.out.println("Done");
        iam.close();
    }

    public static void listAllUsers(IamClient iam) {
        try {
            boolean done = false;
            String newMarker = null;
            while (!done) {
                ListUsersResponse response;
                if (newMarker == null) {
                    ListUsersRequest request =
 ListUsersRequest.builder().build();
                    response = iam.listUsers(request);
                } else {
                    ListUsersRequest request = ListUsersRequest.builder()
                            .marker(newMarker)
                            .build();

                    response = iam.listUsers(request);
                }

                for (User user : response.users()) {
                    System.out.format("\n Retrieved user %s", user.userName());
                    AttachedPermissionsBoundary permissionsBoundary =
 user.permissionsBoundary();
                    if (permissionsBoundary != null)
                        System.out.format("\n Permissions boundary details %s",

 permissionsBoundary.permissionsBoundaryTypeAsString());
                }
```

```
                if (!response.isTruncated()) {
                    done = true;
                } else {
                    newMarker = response.marker();
                }
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les utilisateurs.

```
import { ListUsersCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listUsers = async () => {
  const command = new ListUsersCommand({ MaxItems: 10 });

  const response = await client.send(command);

  for (const { UserName, CreateDate } of response.Users) {
    console.log(`${UserName} created on: ${CreateDate}`);
```

```
  }
  return response;
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 10,
};

iam.listUsers(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var users = data.Users || [];
    users.forEach(function (user) {
      console.log("User " + user.UserName + " created", user.CreateDate);
    });
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun listAllUsers() {
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listUsers(ListUsersRequest { })
        response.users?.forEach { user ->
            println("Retrieved user ${user.userName}")
            val permissionsBoundary = user.permissionsBoundary
            if (permissionsBoundary != null) {
                println("Permissions boundary details
 ${permissionsBoundary.permissionsBoundaryType}")
            }
        }
    }
}
```

- Pour plus de détails sur l'API, consultez ListUsersla section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listUsers($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listUsersArguments = [];
        if ($pathPrefix) {
            $listUsersArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listUsersArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listUsersArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listUsers($listUsersArguments);
    }
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK pour PHP API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple récupère une collection d'utilisateurs dans le fichier actuel Compte AWS.

```
Get-IAMUserList
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:user/Administrator
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE1
UserName         : Administrator
```

```
    Arn              : arn:aws:iam::123456789012:user/Bob
    CreateDate       : 4/6/2015 12:54:42 PM
    PasswordLastUsed : 1/1/0001 12:00:00 AM
    Path             : /
    UserId           : L3EWNONDOM3YUEXAMPLE2
    UserName         : bab

    Arn              : arn:aws:iam::123456789012:user/David
    CreateDate       : 12/10/2014 3:39:27 PM
    PasswordLastUsed : 3/19/2015 8:44:04 AM
    Path             : /
    UserId           : Y4FKWQCXTA52QEXAMPLE3
    UserName         : David
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple récupère une collection d'utilisateurs dans le fichier actuel Compte AWS.

```
Get-IAMUserList
```

Sortie :

```
    Arn              : arn:aws:iam::123456789012:user/Administrator
    CreateDate       : 10/16/2014 9:03:09 AM
    PasswordLastUsed : 3/4/2015 12:12:33 PM
    Path             : /
    UserId           : 7K3GJEANSKZF2EXAMPLE1
    UserName         : Administrator

    Arn              : arn:aws:iam::123456789012:user/Bob
    CreateDate       : 4/6/2015 12:54:42 PM
    PasswordLastUsed : 1/1/0001 12:00:00 AM
    Path             : /
    UserId           : L3EWNONDOM3YUEXAMPLE2
    UserName         : bab

    Arn              : arn:aws:iam::123456789012:user/David
    CreateDate       : 12/10/2014 3:39:27 PM
    PasswordLastUsed : 3/19/2015 8:44:04 AM
```

```
    Path            : /
    UserId          : Y4FKWQCXTA52QEXAMPLE3
    UserName        : David
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_users():
    """
    Lists the users in the current account.

    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
        logger.exception("Couldn't get users.")
        raise
    else:
        return users
```

- Pour plus de détails sur l'API, consultez ListUsersle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# Lists all users in the AWS account
#
# @return [Array<Aws::IAM::Types::User>] An array of user objects
def list_users
  users = []
  @iam_client.list_users.each_page do |page|
    page.users.each do |user|
      users << user
    end
  end
  users
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing users: #{e.message}")
  []
end
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn list_users(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListUsersOutput, SdkError<ListUsersError>> {
    let response = client
        .list_users()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;
    Ok(response)
}
```

- Pour plus de détails sur l'API, voir ListUsersla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_iam->listusers( ).
        MESSAGE 'Retrieved user list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing users.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, consultez ListUsersla section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    public func listUsers() async throws -> [MyUserRecord] {
        var userList: [MyUserRecord] = []

        // Use "Paginated" to get all the users.
        // This lets the SDK handle the 'isTruncated' in "ListUsersOutput".
        let input = ListUsersInput()
        let output = client.listUsersPaginated(input: input)

        do {
            for try await page in output {
                guard let users = page.users else {
                    continue
                }
                for user in users {
                    if let id = user.userId, let name = user.userName {
                        userList.append(MyUserRecord(id: id, name: name))
                    }
                }
            }
        }
        catch {
            print("ERROR: listUsers:", dump(error))
            throw error
        }
       return userList
    }
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`ListVirtualMfaDevices`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `ListVirtualMfaDevices`.

CLI

AWS CLI

Pour afficher la liste des dispositifs MFA virtuels

La commande `list-virtual-mfa-devices` suivante répertorie les dispositifs MFA virtuels qui ont été configurés pour le compte actuel.

```
aws iam list-virtual-mfa-devices
```

Sortie :

```
{
    "VirtualMFADevices": [
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/ExampleMFADevice"
        },
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Fred"
        }
    ]
}
```

Pour de plus d'informations, consultez Activation d'un dispositif virtuel d'authentification multifactorielle (MFA) dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListVirtualMfaDevicesà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple récupère un ensemble de dispositifs MFA virtuels assignés aux utilisateurs du compte. AWS La propriété **`User`** de chaque dispositif est un objet contenant les détails de l'utilisateur IAM auquel le dispositif est attribué.

```
Get-IAMVirtualMFADevice -AssignmentStatus Assigned
```

Sortie :

```
Base32StringSeed :
EnableDate       : 4/13/2015 12:03:42 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/David
User             : Amazon.IdentityManagement.Model.User

Base32StringSeed :
EnableDate       : 4/13/2015 12:06:41 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/root-account-mfa-device
User             : Amazon.IdentityManagement.Model.User
```

- Pour plus de détails sur l'API, reportez-vous ListVirtualMfaDevicesà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple récupère un ensemble de dispositifs MFA virtuels assignés aux utilisateurs du compte. AWS La propriété **`User`** de chaque dispositif est un objet contenant les détails de l'utilisateur IAM auquel le dispositif est attribué.

```
Get-IAMVirtualMFADevice -AssignmentStatus Assigned
```

Sortie :

```
Base32StringSeed :
EnableDate       : 4/13/2015 12:03:42 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/David
```

```
User             : Amazon.IdentityManagement.Model.User

Base32StringSeed :
EnableDate       : 4/13/2015 12:06:41 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/root-account-mfa-device
User             : Amazon.IdentityManagement.Model.User
```

- Pour plus de détails sur l'API, reportez-vous ListVirtualMfaDevicesà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`PutGroupPolicy`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `PutGroupPolicy`.

CLI

AWS CLI

Pour ajouter une politique à un groupe

La commande `put-group-policy` suivante ajoute une politique au groupe IAM nommé `Admins`.

```
aws iam put-group-policy \
    --group-name Admins \
    --policy-document file://AdminPolicy.json \
    --policy-name AdminRoot
```

Cette commande ne produit aucune sortie.

La politique est définie sous la forme d'un document JSON dans le AdminPolicy.jsonfichier. (Le nom et l'extension du fichier n'ont aucune importance.)

Pour plus d'informations, consultez Gestion des politiques IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous PutGroupPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple crée une politique en ligne nommée **AppTesterPolicy** et l'intègre au groupe IAM **AppTesters**. Si une politique en ligne portant le même nom existe déjà, elle est remplacée. Le contenu de la politique JSON provient du fichier **apptesterpolicy.json**. Notez que vous devez utiliser le paramètre **-Raw** pour traiter correctement le contenu du fichier JSON.

```
Write-IAMGroupPolicy -GroupName AppTesters -PolicyName AppTesterPolicy -
PolicyDocument (Get-Content -Raw apptesterpolicy.json)
```

- Pour plus de détails sur l'API, reportez-vous PutGroupPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple crée une politique en ligne nommée **AppTesterPolicy** et l'intègre au groupe IAM **AppTesters**. Si une politique en ligne portant le même nom existe déjà, elle est remplacée. Le contenu de la politique JSON provient du fichier **apptesterpolicy.json**. Notez que vous devez utiliser le paramètre **-Raw** pour traiter correctement le contenu du fichier JSON.

```
Write-IAMGroupPolicy -GroupName AppTesters -PolicyName AppTesterPolicy -
PolicyDocument (Get-Content -Raw apptesterpolicy.json)
```

- Pour plus de détails sur l'API, reportez-vous PutGroupPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **PutRolePermissionsBoundary** avec une CLI

Les exemples de code suivants illustrent comment utiliser `PutRolePermissionsBoundary`.

CLI

AWS CLI

Exemple 1 : pour appliquer une limite des autorisations basée sur une politique personnalisée à un rôle IAM

L'exemple `put-role-permissions-boundary` suivant applique la politique personnalisée nommée `intern-boundary` comme limite des autorisations pour le rôle IAM spécifié.

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
    --role-name lambda-application-role
```

Cette commande ne produit aucune sortie.

Exemple 2 : pour appliquer une limite d'autorisation basée sur une politique AWS gérée à un rôle IAM

L'`put-role-permissions-boundary`exemple suivant applique la `PowerUserAccess` politique AWS gérée comme limite d'autorisations pour le rôle IAM spécifié.

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --role-name x-account-admin
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Modification d'un rôle dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous PutRolePermissionsBoundaryà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple montre comment définir la limite des autorisations pour un rôle IAM. Vous pouvez définir des politiques AWS gérées ou des politiques personnalisées comme limite d'autorisation.

```
Set-IAMRolePermissionsBoundary -RoleName MyRoleName -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Pour plus de détails sur l'API, reportez-vous PutRolePermissionsBoundaryà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple montre comment définir la limite des autorisations pour un rôle IAM. Vous pouvez définir des politiques AWS gérées ou des politiques personnalisées comme limite d'autorisation.

```
Set-IAMRolePermissionsBoundary -RoleName MyRoleName -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Pour plus de détails sur l'API, reportez-vous PutRolePermissionsBoundaryà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`PutRolePolicy`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `PutRolePolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Premiers pas avec les entrepôts de données sans serveur
- Commencer à gérer la configuration
- Débuter avec la protection des appareils liés à l'Internet des objets
- Commencer à utiliser des clusters d'entrepôts de données provisionnés
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous PutRolePolicyà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::putRolePolicy(
        const Aws::String &roleName,
        const Aws::String &policyName,
        const Aws::String &policyDocument,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient iamClient(clientConfig);
    Aws::IAM::Model::PutRolePolicyRequest request;

    request.SetRoleName(roleName);
    request.SetPolicyName(policyName);
    request.SetPolicyDocument(policyDocument);

    Aws::IAM::Model::PutRolePolicyOutcome outcome =
 iamClient.PutRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error putting policy on role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully put the role policy." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, reportez-vous PutRolePolicyà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour attacher une politique d'autorisation à un rôle IAM

La commande `put-role-policy` suivante ajoute une politique d'autorisation au rôle nommé `Test-Role`.

```
aws iam put-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

Cette commande ne produit aucune sortie.

La politique est définie sous la forme d'un document JSON dans le AdminPolicy.jsonfichier. (Le nom et l'extension du fichier n'ont aucune importance.)

Pour attacher une politique d'approbation à un rôle, utilisez la commande `update-assume-role-policy`.

Pour plus d'informations, consultez Modification d'un rôle dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous PutRolePolicyà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { PutRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const examplePolicyDocument = JSON.stringify({
```

```
  Version: "2012-10-17",
  Statement: [
    {
      Sid: "VisualEditor0",
      Effect: "Allow",
      Action: [
        "s3:ListBucketMultipartUploads",
        "s3:ListBucketVersions",
        "s3:ListBucket",
        "s3:ListMultipartUploadParts",
      ],
      Resource: "arn:aws:s3:::amzn-s3-demo-bucket",
    },
    {
      Sid: "VisualEditor1",
      Effect: "Allow",
      Action: [
        "s3:ListStorageLensConfigurations",
        "s3:ListAccessPointsForObjectLambda",
        "s3:ListAllMyBuckets",
        "s3:ListAccessPoints",
        "s3:ListJobs",
        "s3:ListMultiRegionAccessPoints",
      ],
      Resource: "*",
    },
  ],
});

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 * @param {string} policyName
 * @param {string} policyDocument
 */
export const putRolePolicy = async (roleName, policyName, policyDocument) => {
  const command = new PutRolePolicyCommand({
    RoleName: roleName,
    PolicyName: policyName,
    PolicyDocument: policyDocument,
  });
```

```
  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Pour plus de détails sur l'API, reportez-vous PutRolePolicyà la section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple crée une politique en ligne nommée **FedTesterRolePolicy** et l'intègre au rôle IAM. **FedTesterRole** Si une politique en ligne portant le même nom existe déjà, elle est remplacée. Le contenu de la politique JSON provient du fichier **FedTesterPolicy.json**. Notez que vous devez utiliser le paramètre **-Raw** pour traiter correctement le contenu du fichier JSON.

```
Write-IAMRolePolicy -RoleName FedTesterRole -PolicyName FedTesterRolePolicy -
PolicyDocument (Get-Content -Raw FedTesterPolicy.json)
```

- Pour plus de détails sur l'API, reportez-vous PutRolePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple crée une politique en ligne nommée **FedTesterRolePolicy** et l'intègre au rôle IAM. **FedTesterRole** Si une politique en ligne portant le même nom existe déjà, elle est remplacée. Le contenu de la politique JSON provient du fichier **FedTesterPolicy.json**. Notez que vous devez utiliser le paramètre **-Raw** pour traiter correctement le contenu du fichier JSON.

```
Write-IAMRolePolicy -RoleName FedTesterRole -PolicyName FedTesterRolePolicy -
PolicyDocument (Get-Content -Raw FedTesterPolicy.json)
```

- Pour plus de détails sur l'API, reportez-vous PutRolePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`PutUserPermissionsBoundary`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `PutUserPermissionsBoundary`.

CLI

AWS CLI

Exemple 1 : pour appliquer une limite des autorisations basée sur une politique personnalisée à un utilisateur IAM

L'exemple `put-user-permissions-boundary` suivant applique une politique personnalisée nommée `intern-boundary` comme limite des autorisations pour l'utilisateur IAM spécifié.

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
    --user-name intern
```

Cette commande ne produit aucune sortie.

Exemple 2 : pour appliquer une limite d'autorisation basée sur une politique AWS gérée à un utilisateur IAM

L'`put-user-permissions-boundary`exemple suivant applique la politique AWS gérée `PowerUserAccess` nommée limite d'autorisations pour l'utilisateur IAM spécifié.

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --user-name developer
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Ajout et suppression d'autorisations basées sur l'identité IAM du Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous PutUserPermissionsBoundaryà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple montre comment définir la limite des autorisations pour l'utilisateur. Vous pouvez définir des politiques AWS gérées ou des politiques personnalisées comme limite d'autorisation.

```
Set-IAMUserPermissionsBoundary -UserName joe -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Pour plus de détails sur l'API, reportez-vous PutUserPermissionsBoundaryà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple montre comment définir la limite des autorisations pour l'utilisateur. Vous pouvez définir des politiques AWS gérées ou des politiques personnalisées comme limite d'autorisation.

```
Set-IAMUserPermissionsBoundary -UserName joe -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Pour plus de détails sur l'API, reportez-vous PutUserPermissionsBoundaryà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`PutUserPolicy`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `PutUserPolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

CLI

AWS CLI

Pour attacher une politique à un utilisateur IAM

La commande `put-user-policy` suivante attache une politique à l'utilisateur IAM nommé `Bob`.

```
aws iam put-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

Cette commande ne produit aucune sortie.

La politique est définie sous la forme d'un document JSON dans le AdminPolicy.jsonfichier. (Le nom et l'extension du fichier n'ont aucune importance.)

Pour plus d'informations, consultez Ajout et suppression d'autorisations basées sur l'identité IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous PutUserPolicyà la section Référence des AWS CLI commandes.

Go

Kit SDK pour Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"
```

```
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(ctx context.Context, userName string,
 policyName string, actions []string,
 roleArn string) error {
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(ctx, &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
```

```
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

- Pour plus de détails sur l'API, reportez-vous PutUserPolicyà la section Référence des AWS SDK pour Go API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple crée une politique en ligne nommée **EC2AccessPolicy** et l'intègre à l'utilisateur IAM **Bob**. Si une politique en ligne portant le même nom existe déjà, elle est remplacée. Le contenu de la politique JSON provient du fichier **EC2AccessPolicy.json**. Notez que vous devez utiliser le paramètre **-Raw** pour traiter correctement le contenu du fichier JSON.

```
Write-IAMUserPolicy -UserName Bob -PolicyName EC2AccessPolicy -PolicyDocument
 (Get-Content -Raw EC2AccessPolicy.json)
```

- Pour plus de détails sur l'API, reportez-vous PutUserPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple crée une politique en ligne nommée **EC2AccessPolicy** et l'intègre à l'utilisateur IAM **Bob**. Si une politique en ligne portant le même nom existe déjà, elle est remplacée. Le contenu de la politique JSON provient du fichier **EC2AccessPolicy.json**. Notez que vous devez utiliser le paramètre **-Raw** pour traiter correctement le contenu du fichier JSON.

```
Write-IAMUserPolicy -UserName Bob -PolicyName EC2AccessPolicy -PolicyDocument
 (Get-Content -Raw EC2AccessPolicy.json)
```

- Pour plus de détails sur l'API, reportez-vous PutUserPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# Creates an inline policy for a specified user.
# @param username [String] The name of the IAM user.
# @param policy_name [String] The name of the policy to create.
# @param policy_document [String] The JSON policy document.
# @return [Boolean]
def create_user_policy(username, policy_name, policy_document)
  @iam_client.put_user_policy({
                                user_name: username,
                                policy_name: policy_name,
                                policy_document: policy_document
                              })
  @logger.info("Policy #{policy_name} created for user #{username}.")
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Couldn't create policy #{policy_name} for user #{username}.
Here's why:")
  @logger.error("\t#{e.code}: #{e.message}")
  false
end
```

- Pour plus de détails sur l'API, reportez-vous PutUserPolicyà la section Référence des AWS SDK pour Ruby API.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSIAM
import AWSS3


    func putUserPolicy(policyDocument: String, policyName: String, user:
 IAMClientTypes.User) async throws {
        let input = PutUserPolicyInput(
            policyDocument: policyDocument,
            policyName: policyName,
            userName: user.userName
        )
        do {
            _ = try await iamClient.putUserPolicy(input: input)
        } catch {
            print("ERROR: putUserPolicy:", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous PutUserPolicyà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`RemoveClientIdFromOpenIdConnectProvider`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `RemoveClientIdFromOpenIdConnectProvider`.

CLI

AWS CLI

Pour supprimer l'ID client spécifié de la liste des ID client enregistrés pour le fournisseur IAM OpenID Connect spécifié.

Cet exemple supprime l'ID client `My-TestApp-3` de la liste des ID client associés au fournisseur IAM OIDC dont l'ARN est `arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`.

```
aws iam remove-client-id-from-open-id-connect-provider
    --client-id My-TestApp-3 \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Création de fournisseurs d'identité OpenID Connect (OIDC) du Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous RemoveClientIdFromOpenIdConnectProviderà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime l'ID client **`My-TestApp-3`** de la liste des ID clients associés au fournisseur IAM OIDC dont l'ARN est **`arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`**.

```
Remove-IAMClientIDFromOpenIDConnectProvider -ClientID My-TestApp-3
 -OpenIDConnectProviderArn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

- Pour plus de détails sur l'API, reportez-vous RemoveClientIdFromOpenIdConnectProviderà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime l'ID client **My-TestApp-3** de la liste des ID clients associés au fournisseur IAM OIDC dont l'ARN est **arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com**.

```
Remove-IAMClientIDFromOpenIDConnectProvider -ClientID My-TestApp-3
 -OpenIDConnectProviderArn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

- Pour plus de détails sur l'API, reportez-vous RemoveClientIdFromOpenIdConnectProviderà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **RemoveRoleFromInstanceProfile** avec une CLI

Les exemples de code suivants illustrent comment utiliser `RemoveRoleFromInstanceProfile`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Débuter avec le streaming géré
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts

CLI

AWS CLI

Pour supprimer un rôle d'un profil d'instance

La commande `remove-role-from-instance-profile` suivante supprime la politique nommée `Test-Role` du profil d'instance nommé `ExampleInstanceProfile`.

```
aws iam remove-role-from-instance-profile \
    --instance-profile-name ExampleInstanceProfile \
    --role-name Test-Role
```

Pour plus d'informations, consultez Utilisation de profils d'instance dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous RemoveRoleFromInstanceProfileà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime le rôle nommé **MyNewRole** du profil d'instance EC2 nommé **MyNewRole**. Un profil d'instance créé dans la console IAM porte toujours le même nom que le rôle, comme dans cet exemple. Si vous les créez dans l'API ou la CLI, ils peuvent porter des noms différents.

```
Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyNewRole -RoleName
 MyNewRole -Force
```

- Pour plus de détails sur l'API, reportez-vous RemoveRoleFromInstanceProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime le rôle nommé **MyNewRole** du profil d'instance EC2 nommé **MyNewRole**. Un profil d'instance créé dans la console IAM porte toujours le même nom que le rôle, comme dans cet exemple. Si vous les créez dans l'API ou la CLI, ils peuvent porter des noms différents.

```
Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyNewRole -RoleName
 MyNewRole -Force
```

- Pour plus de détails sur l'API, reportez-vous RemoveRoleFromInstanceProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`RemoveUserFromGroup`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `RemoveUserFromGroup`.

CLI

AWS CLI

Pour supprimer un utilisateur d'un groupe IAM

La commande `remove-user-from-group` suivante supprime l'utilisateur nommé Bob du groupe IAM nommé `Admins`.

```
aws iam remove-user-from-group \
    --user-name Bob \
    --group-name Admins
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Ajout et suppression d'utilisateurs dans un groupe d'utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous RemoveUserFromGroupà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime l'utilisateur IAM **`Bob`** du groupe **`Testers`**.

```
Remove-IAMUserFromGroup -GroupName Testers -UserName Bob
```

Exemple 2 : cet exemple trouve tous les groupes dont l'utilisateur IAM **`Theresa`** est membre, puis retire **`Theresa`** de ces groupes.

```
$groups = Get-IAMGroupForUser -UserName Theresa
```

```
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName Theresa -Force }
```

Exemple 3 : cet exemple montre une autre méthode pour supprimer l'utilisateur IAM **Bob** du groupe **Testers**.

```
Get-IAMGroupForUser -UserName Bob | Remove-IAMUserFromGroup -UserName Bob -
GroupName Testers -Force
```

- Pour plus de détails sur l'API, reportez-vous RemoveUserFromGroupà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime l'utilisateur IAM **Bob** du groupe **Testers**.

```
Remove-IAMUserFromGroup -GroupName Testers -UserName Bob
```

Exemple 2 : cet exemple trouve tous les groupes dont l'utilisateur IAM **Theresa** est membre, puis retire **Theresa** de ces groupes.

```
$groups = Get-IAMGroupForUser -UserName Theresa
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName Theresa -Force }
```

Exemple 3 : cet exemple montre une autre méthode pour supprimer l'utilisateur IAM **Bob** du groupe **Testers**.

```
Get-IAMGroupForUser -UserName Bob | Remove-IAMUserFromGroup -UserName Bob -
GroupName Testers -Force
```

- Pour plus de détails sur l'API, reportez-vous RemoveUserFromGroupà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **ResyncMfaDevice** avec une CLI

Les exemples de code suivants illustrent comment utiliser `ResyncMfaDevice`.

CLI

AWS CLI

Pour resynchroniser un dispositif MFA

L'exemple `resync-mfa-device` suivant synchronise le dispositif MFA associé à l'utilisateur IAM Bob, et dont l'ARN est `arn:aws:iam::123456789012:mfa/BobsMFADevice`, avec un programme d'authentification qui a fourni les deux codes d'authentification.

```
aws iam resync-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 987654
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Utilisation de l'authentification multifactorielle (MFA) dans AWS dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ResyncMfaDeviceà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple synchronise le dispositif MFA associé à l'utilisateur IAM **Bob** et dont l'ARN est **arn:aws:iam::123456789012:mfa/bob** avec un programme d'authentification qui a fourni les deux codes d'authentification.

```
Sync-IAMMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/theresa -
AuthenticationCode1 123456 -AuthenticationCode2 987654 -UserName Bob
```

Exemple 2 : cet exemple synchronise le dispositif MFA IAM associé à l'utilisateur IAM **Theresa** avec un périphérique physique possédant le numéro de série **ABCD12345678** et qui a fourni les deux codes d'authentification.

```
Sync-IAMMFADevice -SerialNumber ABCD12345678 -AuthenticationCode1 123456 -
AuthenticationCode2 987654 -UserName Theresa
```

- Pour plus de détails sur l'API, reportez-vous ResyncMfaDeviceà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple synchronise le dispositif MFA associé à l'utilisateur IAM **Bob** et dont l'ARN est **arn:aws:iam::123456789012:mfa/bob** avec un programme d'authentification qui a fourni les deux codes d'authentification.

```
Sync-IAMMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/theresa -
AuthenticationCode1 123456 -AuthenticationCode2 987654 -UserName Bob
```

Exemple 2 : cet exemple synchronise le dispositif MFA IAM associé à l'utilisateur IAM **Theresa** avec un périphérique physique possédant le numéro de série **ABCD12345678** et qui a fourni les deux codes d'authentification.

```
Sync-IAMMFADevice -SerialNumber ABCD12345678 -AuthenticationCode1 123456 -
AuthenticationCode2 987654 -UserName Theresa
```

- Pour plus de détails sur l'API, reportez-vous ResyncMfaDeviceà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **SetDefaultPolicyVersion**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `SetDefaultPolicyVersion`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Gérer les politiques
- Restaurer une version de stratégie

CLI

AWS CLI

Pour définir la version spécifiée de la politique spécifiée comme la version par défaut de la politique.

Cet exemple définit la version v2 de la politique dont l'ARN est `arn:aws:iam::123456789012:policy/MyPolicy` en tant que version active par défaut.

```
aws iam set-default-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Pour plus d'informations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir SetDefaultPolicyVersionla section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple définit la version **v2** de la politique dont l'ARN est **arn:aws:iam::123456789012:policy/MyPolicy** en tant que version active par défaut.

```
Set-IAMDefaultPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy
 -VersionId v2
```

- Pour plus de détails sur l'API, reportez-vous SetDefaultPolicyVersionà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple définit la version **v2** de la politique dont l'ARN est **arn:aws:iam::123456789012:policy/MyPolicy** en tant que version active par défaut.

```
Set-IAMDefaultPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy
 -VersionId v2
```

- Pour plus de détails sur l'API, reportez-vous SetDefaultPolicyVersionà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    lo_iam->setdefaultpolicyversion(
      iv_policyarn = iv_policy_arn
      iv_versionid = iv_version_id ).
    MESSAGE 'Default policy version set successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Policy or version does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iaminvalidinputex.
    MESSAGE 'Invalid input provided.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Limit exceeded.' TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, consultez SetDefaultPolicyVersionla section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`TagRole`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `TagRole`.

CLI

AWS CLI

Pour ajouter une balise à un rôle

La commande `tag-role` suivante ajoute une balise avec un nom de département au rôle spécifié.

```
aws iam tag-role --role-name my-role \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Étiquetage de ressources IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous TagRoleà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple ajoute une balise au rôle dans le service de gestion des identités

```
Add-IAMRoleTag -RoleName AdminRoleacess -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Pour plus de détails sur l'API, reportez-vous TagRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple ajoute une balise au rôle dans le service de gestion des identités

```
Add-IAMRoleTag -RoleName AdminRoleacess -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Pour plus de détails sur l'API, reportez-vous TagRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`TagUser`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `TagUser`.

CLI

AWS CLI

Pour ajouter une balise à un utilisateur

La commande `tag-user` suivante ajoute une balise avec le département associé à l'utilisateur spécifié.

```
aws iam tag-user \
    --user-name alice \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Étiquetage de ressources IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous TagUserà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple ajoute une balise à l'utilisateur dans le service de gestion des identités

```
Add-IAMUserTag -UserName joe -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Pour plus de détails sur l'API, reportez-vous TagUserà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple ajoute une balise à l'utilisateur dans le service de gestion des identités

```
Add-IAMUserTag -UserName joe -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Pour plus de détails sur l'API, reportez-vous TagUserà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`UntagRole`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `UntagRole`.

CLI

AWS CLI

Pour supprimer une balise d'un rôle

La commande `untag-role` suivante supprime toute balise portant le nom de clé « Department » du rôle spécifié.

```
aws iam untag-role \
    --role-name my-role \
    --tag-keys Department
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Étiquetage de ressources IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UntagRoleà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple supprime la balise du rôle nommé « MyRoleName » avec la clé de balise « abac ». Pour supprimer plusieurs balises, fournissez une liste de clés de balises séparées par des virgules.

```
Remove-IAMRoleTag -RoleName MyRoleName -TagKey "abac","xyzw"
```

- Pour plus de détails sur l'API, reportez-vous UntagRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple supprime la balise du rôle nommé « MyRoleName » avec la clé de balise « abac ». Pour supprimer plusieurs balises, fournissez une liste de clés de balises séparées par des virgules.

```
Remove-IAMRoleTag -RoleName MyRoleName -TagKey "abac","xyzw"
```

- Pour plus de détails sur l'API, reportez-vous UntagRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`UntagUser`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `UntagUser`.

CLI

AWS CLI

Pour supprimer une balise d'un utilisateur

La commande `untag-user` suivante supprime toute balise portant le nom de clé « Department » de l'utilisateur spécifié.

```
aws iam untag-user \
    --user-name alice \
    --tag-keys Department
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Étiquetage de ressources IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UntagUserà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple supprime la balise de l'utilisateur nommée « joe » dont les clés de balise sont « abac » et « xyzw ». Pour supprimer plusieurs balises, fournissez une liste de clés de balises séparées par des virgules.

```
Remove-IAMUserTag -UserName joe -TagKey "abac","xyzw"
```

- Pour plus de détails sur l'API, reportez-vous UntagUserà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple supprime la balise de l'utilisateur nommée « joe » dont les clés de balise sont « abac » et « xyzw ». Pour supprimer plusieurs balises, fournissez une liste de clés de balises séparées par des virgules.

```
Remove-IAMUserTag -UserName joe -TagKey "abac","xyzw"
```

- Pour plus de détails sur l'API, reportez-vous UntagUserà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`UpdateAccessKey`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `UpdateAccessKey`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer des clés d'accès

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function iam_update_access_key
#
# This function can activate or deactivate an IAM access key for the specified
 IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to update.
#       -a            -- Activate the selected access key.
#       -d            -- Deactivate the selected access key.
#
# Example:
#       # To deactivate the selected access key for IAM user Bob
#       iam_update_access_key -u Bob -k AKIAIOSFODNN7EXAMPLE -d
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_update_access_key() {
  local user_name access_key status response
```

```
 local option OPTARG # Required to use getopts command in a function.
 local activate_flag=false deactivate_flag=false

 # bashsupport disable=BP5008
 function usage() {
   echo "function iam_update_access_key"
   echo "Updates the status of an AWS Identity and Access Management (IAM)
access key for the specified IAM user"
   echo "  -u user_name    The name of the user."
   echo "  -k access_key   The access key to update."
   echo "  -a              Activate the access key."
   echo "  -d              Deactivate the access key."
   echo ""
 }

 # Retrieve the calling parameters.
   while getopts "u:k:adh" option; do
     case "${option}" in
       u) user_name="${OPTARG}" ;;
       k) access_key="${OPTARG}" ;;
       a) activate_flag=true ;;
       d) deactivate_flag=true ;;
       h)
         usage
         return 0
         ;;
       \?)
         echo "Invalid parameter"
         usage
         return 1
         ;;
     esac
   done
   export OPTIND=1

  # Validate input parameters
   if [[ -z "$user_name" ]]; then
     errecho "ERROR: You must provide a username with the -u parameter."
     usage
     return 1
   fi

   if [[ -z "$access_key" ]]; then
     errecho "ERROR: You must provide an access key with the -k parameter."
```

```
      usage
      return 1
    fi

    # Ensure that only -a or -d is specified
    if [[ "$activate_flag" == true && "$deactivate_flag" == true ]]; then
      errecho "ERROR: You cannot specify both -a (activate) and -d (deactivate)
at the same time."
      usage
      return 1
    fi

    # If neither -a nor -d is provided, return an error
    if [[ "$activate_flag" == false && "$deactivate_flag" == false ]]; then
      errecho "ERROR: You must specify either -a (activate) or -d (deactivate)."
      usage
      return 1
    fi

    # Determine the status based on the flag
    if [[ "$activate_flag" == true ]]; then
      status="Active"
    elif [[ "$deactivate_flag" == true ]]; then
      status="Inactive"
    fi

    iecho "Parameters:\n"
    iecho "    Username:   $user_name"
    iecho "    Access key: $access_key"
    iecho "    New status: $status"
    iecho ""

    # Update the access key status
    response=$(aws iam update-access-key \
      --user-name "$user_name" \
      --access-key-id "$access_key" \
      --status "$status" 2>&1)

    local error_code=${?}

    if [[ $error_code -ne 0 ]]; then
      aws_cli_error_log $error_code
      errecho "ERROR: AWS reports update-access-key operation failed.\n$response"
      return 1
```

```
    fi

    iecho "update-access-key response: $response"
    iecho

    return 0
}
```

- Pour plus de détails sur l'API, reportez-vous UpdateAccessKeyà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::updateAccessKey(const Aws::String &userName,
                                  const Aws::String &accessKeyID,
                                  Aws::IAM::Model::StatusType status,
                                  const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::UpdateAccessKeyRequest request;
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);
    request.SetStatus(status);

    auto outcome = iam.UpdateAccessKey(request);
    if (outcome.IsSuccess()) {
        std::cout << "Successfully updated status of access key "
                  << accessKeyID << " for user " << userName << std::endl;
    }
    else {
        std::cerr << "Error updated status of access key " << accessKeyID <<
                  " for user " << userName << ": " <<
```

```
                    outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, reportez-vous UpdateAccessKeyà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour activer ou désactiver une clé d'accès pour un utilisateur IAM

La commande `update-access-key` suivante désactive la clé d'accès spécifiée (un ID de clé d'accès rapide et une clé d'accès secrète) pour l'utilisateur IAM nommé Bob.

```
aws iam update-access-key \
    --access-key-id AKIAIOSFODNN7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

Cette commande ne produit aucune sortie.

La désactivation de la clé signifie qu'elle ne peut pas être utilisée pour un accès programmatique à. AWS Cependant, la clé est toujours disponible et peut être réactivée.

Pour plus d'informations, consultez Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateAccessKeyà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.StatusType;
import software.amazon.awssdk.services.iam.model.UpdateAccessKeyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateAccessKey {

    private static StatusType statusType;

    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username> <accessId> <status>\s

                Where:
                    username - The name of the user whose key you want to update.
\s
                    accessId - The access key ID of the secret access key you
 want to update.\s
                    status - The status you want to assign to the secret access
 key.\s
```

```
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        String accessId = args[1];
        String status = args[2];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateKey(iam, username, accessId, status);
        System.out.println("Done");
        iam.close();
    }

    public static void updateKey(IamClient iam, String username, String accessId,
String status) {
        try {
            if (status.toLowerCase().equalsIgnoreCase("active")) {
                statusType = StatusType.ACTIVE;
            } else if (status.toLowerCase().equalsIgnoreCase("inactive")) {
                statusType = StatusType.INACTIVE;
            } else {
                statusType = StatusType.UNKNOWN_TO_SDK_VERSION;
            }

            UpdateAccessKeyRequest request = UpdateAccessKeyRequest.builder()
                    .accessKeyId(accessId)
                    .userName(username)
                    .status(statusType)
                    .build();

            iam.updateAccessKey(request);
            System.out.printf("Successfully updated the status of access key %s
to" +
                    "status %s for user %s", accessId, status, username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
```

```
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous UpdateAccessKeyà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Mettez à jour la clé d'accès.

```
import {
  UpdateAccessKeyCommand,
  IAMClient,
  StatusType,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
 */
export const updateAccessKey = (userName, accessKeyId) => {
  const command = new UpdateAccessKeyCommand({
    AccessKeyId: accessKeyId,
    Status: StatusType.Inactive,
    UserName: userName,
  });

  return client.send(command);
```

```
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous UpdateAccessKeyà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  AccessKeyId: "ACCESS_KEY_ID",
  Status: "Active",
  UserName: "USER_NAME",
};

iam.updateAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous UpdateAccessKeyà la section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple modifie le statut de la clé d'accès **AKIAIOSFODNN7EXAMPLE**, pour l'utilisateur IAM nommé **Bob**, à **Inactive**.

```
Update-IAMAccessKey -UserName Bob -AccessKeyId AKIAIOSFODNN7EXAMPLE -Status
 Inactive
```

- Pour plus de détails sur l'API, reportez-vous UpdateAccessKeyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple modifie le statut de la clé d'accès **AKIAIOSFODNN7EXAMPLE**, pour l'utilisateur IAM nommé **Bob**, à **Inactive**.

```
Update-IAMAccessKey -UserName Bob -AccessKeyId AKIAIOSFODNN7EXAMPLE -Status
 Inactive
```

- Pour plus de détails sur l'API, reportez-vous UpdateAccessKeyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def update_key(user_name, key_id, activate):
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
```

```
    :param activate: When True, the key is activated. Otherwise, the key is
deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
        if activate:
            key.activate()
        else:
            key.deactivate()
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
key_id)
    except ClientError:
        logger.exception(
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
key_id
        )
        raise
```

- Pour plus de détails sur l'API, consultez UpdateAccessKeyle AWS manuel de référence de l'API SDK for Python (Boto3).

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        lo_iam->updateaccesskey(
          iv_accesskeyid = iv_access_key_id
          iv_status = iv_status
          iv_username = iv_user_name ).
        MESSAGE 'Access key updated successfully.' TYPE 'I'.
```

```
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Access key or user does not exist.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous UpdateAccessKeyà la section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`UpdateAccountPasswordPolicy`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `UpdateAccountPasswordPolicy`.

CLI

AWS CLI

Pour définir ou modifier la politique de mot de passe du compte actuel

La commande `update-account-password-policy` suivante définit la politique de mot de passe pour exiger une longueur minimale de huit caractères et un ou plusieurs chiffres dans le mot de passe.

```
aws iam update-account-password-policy \
    --minimum-password-length 8 \
    --require-numbers
```

Cette commande ne produit aucune sortie.

Les modifications apportées à la politique de mot de passe d'un compte affectent tous les nouveaux mots de passe créés pour les utilisateurs IAM du compte. Les modifications apportées à la politique des mots de passe n'affectent pas les mots de passe existants.

Pour plus d'informations, consultez Définition d'une politique de mot de passe du compte pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateAccountPasswordPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple met à jour la politique de mot de passe du compte avec les paramètres spécifiés. Notez que les paramètres qui ne sont pas inclus dans la commande ne sont pas laissés inchangés. Au lieu de cela, ils sont réinitialisés aux valeurs par défaut.

```
Update-IAMAccountPasswordPolicy -AllowUsersToChangePasswords $true -HardExpiry
 $false -MaxPasswordAge 90 -MinimumPasswordLength 8 -PasswordReusePrevention 20
 -RequireLowercaseCharacters $true -RequireNumbers $true -RequireSymbols $true -
RequireUppercaseCharacters $true
```

- Pour plus de détails sur l'API, reportez-vous UpdateAccountPasswordPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple met à jour la politique de mot de passe du compte avec les paramètres spécifiés. Notez que les paramètres qui ne sont pas inclus dans la commande ne sont pas laissés inchangés. Au lieu de cela, ils sont réinitialisés aux valeurs par défaut.

```
Update-IAMAccountPasswordPolicy -AllowUsersToChangePasswords $true -HardExpiry
 $false -MaxPasswordAge 90 -MinimumPasswordLength 8 -PasswordReusePrevention 20
 -RequireLowercaseCharacters $true -RequireNumbers $true -RequireSymbols $true -
RequireUppercaseCharacters $true
```

- Pour plus de détails sur l'API, reportez-vous UpdateAccountPasswordPolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`UpdateAssumeRolePolicy`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `UpdateAssumeRolePolicy`.

CLI

AWS CLI

Pour mettre à jour la politique d'approbation d'un rôle IAM

La commande `update-assume-role-policy` suivante met à jour la politique d'approbation pour le rôle nommé `Test-Role`.

```
aws iam update-assume-role-policy \
    --role-name Test-Role \
    --policy-document file://Test-Role-Trust-Policy.json
```

Cette commande ne produit aucune sortie.

La politique de confiance est définie sous la forme d'un document JSON dans le Test-Role-Trust-Policy.jsonfichier. (Le nom et l'extension du fichier n'ont aucune importance.) La politique d'approbation doit spécifier un principal.

Pour mettre à jour la politique des autorisations d'un rôle, utilisez la commande `put-role-policy`.

Pour plus d'informations, consultez Création de rôles IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateAssumeRolePolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple met à jour le rôle IAM nommé **`ClientRole`** avec une nouvelle politique d'approbation dont le contenu provient du fichier **`ClientRolePolicy.json`**. Notez que vous devez utiliser le paramètre booléen **`-Raw`** pour traiter correctement le contenu du fichier JSON.

```
Update-IAMAssumeRolePolicy -RoleName ClientRole -PolicyDocument (Get-Content -raw
 ClientRolePolicy.json)
```

- Pour plus de détails sur l'API, reportez-vous UpdateAssumeRolePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple met à jour le rôle IAM nommé **`ClientRole`** avec une nouvelle politique d'approbation dont le contenu provient du fichier **`ClientRolePolicy.json`**. Notez que vous devez utiliser le paramètre booléen **`-Raw`** pour traiter correctement le contenu du fichier JSON.

```
Update-IAMAssumeRolePolicy -RoleName ClientRole -PolicyDocument (Get-Content -raw
 ClientRolePolicy.json)
```

- Pour plus de détails sur l'API, reportez-vous UpdateAssumeRolePolicyà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`UpdateGroup`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `UpdateGroup`.

CLI

AWS CLI

Pour renommer un groupe IAM

La commande `update-group` suivante remplace le nom du groupe IAM `Test` par `Test-1`.

```
aws iam update-group \
    --group-name Test \
    --new-group-name Test-1
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Affectation d'un nouveau nom à un groupe IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateGroupà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renomme le groupe IAM **Testers** en **AppTesters**.

```
Update-IAMGroup -GroupName Testers -NewGroupName AppTesters
```

Exemple 2 : dans cet exemple, le chemin du groupe IAM **AppTesters** est remplacé par **/Org1/Org2/**. Cela change l'ARN du groupe en **arn:aws:iam::123456789012:group/Org1/Org2/AppTesters**.

```
Update-IAMGroup -GroupName AppTesters -NewPath /Org1/Org2/
```

- Pour plus de détails sur l'API, reportez-vous UpdateGroupà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renomme le groupe IAM **Testers** en **AppTesters**.

```
Update-IAMGroup -GroupName Testers -NewGroupName AppTesters
```

Exemple 2 : dans cet exemple, le chemin du groupe IAM **AppTesters** est remplacé par **/Org1/Org2/**. Cela change l'ARN du groupe en **arn:aws:iam::123456789012:group/Org1/Org2/AppTesters**.

```
Update-IAMGroup -GroupName AppTesters -NewPath /Org1/Org2/
```

- Pour plus de détails sur l'API, reportez-vous UpdateGroupà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`UpdateLoginProfile`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `UpdateLoginProfile`.

CLI

AWS CLI

Pour mettre à jour le mot de passe d'un utilisateur IAM

Par exemple, la commande suivante crée un utilisateur IAM nommé `update-login-profile` avec le mot de passe `Bob`.

```
aws iam update-login-profile \
    --user-name Bob \
    --password <password>
```

Cette commande ne produit aucune sortie.

Pour définir une politique de mot de passe pour le compte, utilisez la commande `update-account-password-policy`. Si le nouveau mot de passe enfreint la politique de mot de passe du compte, la commande renvoie une erreur `PasswordPolicyViolation`.

Si la politique de mot de passe du compte le permet, les utilisateurs IAM peuvent modifier leurs propres mots de passe à l'aide de la commande `change-password`.

Conservez le nouveau mot de passe en lieu sûr. Si le mot de passe est perdu, il ne peut pas être récupéré et vous devez recréer un à l'aide de la commande `create-login-profile`.

Pour plus d'informations, consultez Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateLoginProfileà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple définit un nouveau mot de passe temporaire pour l'utilisateur IAM **`Bob`** et demande à ce dernier de le modifier lors de sa prochaine connexion.

```
Update-IAMLoginProfile -UserName Bob -Password "P@ssw0rd1234" -
PasswordResetRequired $true
```

- Pour plus de détails sur l'API, reportez-vous UpdateLoginProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple définit un nouveau mot de passe temporaire pour l'utilisateur IAM **Bob** et demande à ce dernier de le modifier lors de sa prochaine connexion.

```
Update-IAMLoginProfile -UserName Bob -Password "P@ssw0rd1234" -
PasswordResetRequired $true
```

- Pour plus de détails sur l'API, reportez-vous UpdateLoginProfileà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`UpdateOpenIdConnectProviderThumbprint`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `UpdateOpenIdConnectProviderThumbprint`.

CLI

AWS CLI

Pour remplacer la liste existante des empreintes numériques des certificats de serveur par une nouvelle liste

Cet exemple met à jour la liste des empreintes numériques des certificats pour le fournisseur OIDC dont l'ARN est `arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com` afin d'utiliser une nouvelle empreinte.

```
aws iam update-open-id-connect-provider-thumbprint \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com \
    --thumbprint-list 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

Si les points de terminaison JWKS et de découverte de votre fournisseur OIDC utilisent des certificats différents, incluez les empreintes digitales des deux points de terminaison dans la valeur. `--thumbprint-list`

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Création de fournisseurs d'identité OpenID Connect (OIDC) du Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateOpenIdConnectProviderThumbprintà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple met à jour la liste des empreintes numériques des certificats pour le fournisseur OIDC dont l'ARN est **`arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`** afin d'utiliser une nouvelle empreinte. Le fournisseur OIDC partage la nouvelle valeur lorsque le certificat associé au fournisseur change.

```
Update-IAMOpenIDConnectProviderThumbprint -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com -ThumbprintList
 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

- Pour plus de détails sur l'API, reportez-vous UpdateOpenIdConnectProviderThumbprintà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple met à jour la liste des empreintes numériques des certificats pour le fournisseur OIDC dont l'ARN est **`arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`** afin d'utiliser une nouvelle empreinte. Le fournisseur OIDC partage la nouvelle valeur lorsque le certificat associé au fournisseur change.

```
Update-IAMOpenIDConnectProviderThumbprint -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com -ThumbprintList
 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

- Pour plus de détails sur l'API, reportez-vous UpdateOpenIdConnectProviderThumbprintà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`UpdateRole`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `UpdateRole`.

CLI

AWS CLI

Pour modifier la description ou la durée de session d'un rôle IAM

La commande `update-role` suivante remplace la description du rôle IAM `production-role` par `Main production role` et définit la durée maximale de session à 12 heures.

```
aws iam update-role \
    --role-name production-role \
    --description 'Main production role' \
    --max-session-duration 43200
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Modification d'un rôle dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateRoleà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple met à jour la description du rôle et la durée maximale de session (en secondes) pour laquelle la session d'un rôle peut être demandée.

```
Update-IAMRole -RoleName MyRoleName -Description "My testing role" -
MaxSessionDuration 43200
```

- Pour plus de détails sur l'API, reportez-vous UpdateRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple met à jour la description du rôle et la durée maximale de session (en secondes) pour laquelle la session d'un rôle peut être demandée.

```
Update-IAMRole -RoleName MyRoleName -Description "My testing role" -
MaxSessionDuration 43200
```

- Pour plus de détails sur l'API, reportez-vous UpdateRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`UpdateRoleDescription`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `UpdateRoleDescription`.

CLI

AWS CLI

Pour modifier la description d'un rôle IAM

La commande `update-role` suivante remplace le nom du rôle IAM `production-role` par `Main production role`.

```
aws iam update-role-description \
    --role-name production-role \
    --description 'Main production role'
```

Sortie :

```
{
    "Role": {
        "Path": "/",
        "RoleName": "production-role",
        "RoleId": "AROA1234567890EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/production-role",
        "CreateDate": "2017-12-06T17:16:37+00:00",
```

```
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole",
                    "Condition": {}
                }
            ]
        },
        "Description": "Main production role"
    }
}
```

Pour plus d'informations, consultez Modification d'un rôle dans le AWS Guide de l'utilisateur IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateRoleDescriptionà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple met à jour la description d'un rôle IAM dans votre compte.

```
Update-IAMRoleDescription -RoleName MyRoleName -Description "My testing role"
```

- Pour plus de détails sur l'API, reportez-vous UpdateRoleDescriptionà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple met à jour la description d'un rôle IAM dans votre compte.

```
Update-IAMRoleDescription -RoleName MyRoleName -Description "My testing role"
```

- Pour plus de détails sur l'API, reportez-vous UpdateRoleDescriptionà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`UpdateSamlProvider`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `UpdateSamlProvider`.

CLI

AWS CLI

Pour mettre à jour le document de métadonnées d'un fournisseur SAML existant

Cet exemple met à jour le fournisseur SAML dans IAM, dont l'ARN est `arn:aws:iam::123456789012:saml-provider/SAMLADFS`, avec un nouveau document de métadonnées SAML issu du fichier `SAMLMetaData.xml`.

```
aws iam update-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

Sortie :

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/SAMLADFS"
}
```

Pour plus d'informations, consultez Création de fournisseurs d'identité SAML IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateSamlProviderà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple met à jour le fournisseur SAML dans IAM, dont l'ARN est **`arn:aws:iam::123456789012:saml-provider/SAMLADFS`**, avec un nouveau document

de métadonnées SAML issu du fichier **`SAMLMetaData.xml`**. Notez que vous devez utiliser le paramètre booléen **`-Raw`** pour traiter correctement le contenu du fichier JSON.

```
Update-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS -SAMLMetadataDocument (Get-Content -Raw SAMLMetaData.xml)
```

- Pour plus de détails sur l'API, reportez-vous UpdateSamlProviderà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple met à jour le fournisseur SAML dans IAM, dont l'ARN est **`arn:aws:iam::123456789012:saml-provider/SAMLADFS`**, avec un nouveau document de métadonnées SAML issu du fichier **`SAMLMetaData.xml`**. Notez que vous devez utiliser le paramètre booléen **`-Raw`** pour traiter correctement le contenu du fichier JSON.

```
Update-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS -SAMLMetadataDocument (Get-Content -Raw SAMLMetaData.xml)
```

- Pour plus de détails sur l'API, reportez-vous UpdateSamlProviderà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`UpdateServerCertificate`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `UpdateServerCertificate`.

C++

Kit de développement logiciel (SDK) for C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::updateServerCertificate(const Aws::String
 &currentCertificateName,
                                           const Aws::String &newCertificateName,
                                           const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::UpdateServerCertificateRequest request;
    request.SetServerCertificateName(currentCertificateName);
    request.SetNewServerCertificateName(newCertificateName);

    auto outcome = iam.UpdateServerCertificate(request);
    bool result = true;
    if (outcome.IsSuccess()) {
        std::cout << "Server certificate " << currentCertificateName
                  << " successfully renamed as " << newCertificateName
                  << std::endl;
    }
    else {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error changing name of server certificate " <<
                      currentCertificateName << " to " << newCertificateName <<
 ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << currentCertificateName
                      << "' not found." << std::endl;
        }
    }

    return result;
}
```

- Pour plus de détails sur l'API, reportez-vous UpdateServerCertificateà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour modifier le chemin ou le nom d'un certificat de serveur dans votre AWS compte

La commande `update-server-certificate` suivante fait passer le nom du certificat de `myServerCertificate` à `myUpdatedServerCertificate`. Il modifie également le chemin pour `/cloudfront/` qu'il soit accessible par le CloudFront service Amazon. Cette commande ne produit aucune sortie. Vous pouvez afficher les résultats de la mise à jour en exécutant la commande `list-server-certificates`.

```
aws-iam update-server-certificate \
    --server-certificate-name myServerCertificate \
    --new-server-certificate-name myUpdatedServerCertificate \
    --new-path /cloudfront/
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Gestion des certificats de serveur dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateServerCertificateà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Mettez à jour un certificat de serveur.

```
import { UpdateServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});
```

```
/**
 *
 * @param {string} currentName
 * @param {string} newName
 */
export const updateServerCertificate = (currentName, newName) => {
  const command = new UpdateServerCertificateCommand({
    ServerCertificateName: currentName,
    NewServerCertificateName: newName,
  });

  return client.send(command);
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous UpdateServerCertificateà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  ServerCertificateName: "CERTIFICATE_NAME",
  NewServerCertificateName: "NEW_CERTIFICATE_NAME",
};

iam.updateServerCertificate(params, function (err, data) {
```

```
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, reportez-vous UpdateServerCertificateà la section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renomme le certificat nommé **MyServerCertificate** en **MyRenamedServerCertificate**.

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -
NewServerCertificateName MyRenamedServerCertificate
```

Exemple 2 : Cet exemple déplace le certificat nommé **MyServerCertificate** vers le chemin/Org1/Org2/. Cela change l'ARN de la ressource en **arn:aws:iam::123456789012:server-certificate/Org1/Org2/MyServerCertificate**.

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -NewPath /
Org1/Org2/
```

- Pour plus de détails sur l'API, reportez-vous UpdateServerCertificateà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renomme le certificat nommé **MyServerCertificate** en **MyRenamedServerCertificate**.

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -
NewServerCertificateName MyRenamedServerCertificate
```

Exemple 2 : Cet exemple déplace le certificat nommé **MyServerCertificate** vers le chemin/Org1/Org2/. Cela change l'ARN de la ressource en **arn:aws:iam::123456789012:server-certificate/Org1/Org2/MyServerCertificate**.

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -NewPath /
Org1/Org2/
```

- Pour plus de détails sur l'API, reportez-vous UpdateServerCertificateà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez, mettez à jour et supprimez des certificats de serveur.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'ServerCertificateManager'
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key
                                          })
```

```
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info('No server certificates found.')
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
#{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
'#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
```

```
    false
  end
end
```

- Pour plus de détails sur l'API, reportez-vous UpdateServerCertificateà la section Référence des AWS SDK pour Ruby API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`UpdateSigningCertificate`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `UpdateSigningCertificate`.

CLI

AWS CLI

Pour activer ou désactiver un certificat de signature pour un utilisateur IAM

La commande `update-signing-certificate` suivante désactive le certificat de signature spécifié pour l'utilisateur IAM nommé Bob.

```
aws iam update-signing-certificate \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

Pour obtenir l'ID d'un certificat de signature, utilisez la commande `list-signing-certificates`.

Pour plus d'informations, consultez Gestion des certificats de signature dans le Guide de l'utilisateur Amazon EC2.

- Pour plus de détails sur l'API, reportez-vous UpdateSigningCertificateà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple met à jour le certificat associé à l'utilisateur IAM nommé **Bob** et dont l'ID de certificat est **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** pour le marquer comme inactif.

```
Update-IAMSigningCertificate -CertificateId Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU -
UserName Bob -Status Inactive
```

- Pour plus de détails sur l'API, reportez-vous UpdateSigningCertificateà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple met à jour le certificat associé à l'utilisateur IAM nommé **Bob** et dont l'ID de certificat est **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** pour le marquer comme inactif.

```
Update-IAMSigningCertificate -CertificateId Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU -
UserName Bob -Status Inactive
```

- Pour plus de détails sur l'API, reportez-vous UpdateSigningCertificateà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`UpdateUser`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `UpdateUser`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Créer des utilisateurs en lecture seule et en lecture-écriture

C++

SDK pour C++

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::updateUser(const Aws::String &currentUserName,
                             const Aws::String &newUserName,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::UpdateUserRequest request;
    request.SetUserName(currentUserName);
    request.SetNewUserName(newUserName);

    auto outcome = iam.UpdateUser(request);
    if (outcome.IsSuccess()) {
        std::cout << "IAM user " << currentUserName <<
                  " successfully updated with new user name " << newUserName <<
                  std::endl;
    }
    else {
        std::cerr << "Error updating user name for IAM user " << currentUserName
 <<
                  ":" << outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, voir UpdateUserla section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour modifier le nom d'un utilisateur IAM

La commande `update-user` suivante fait passer le nom de l'utilisateur IAM de `Bob` à `Robert`.

```
aws iam update-user \
    --user-name Bob \
    --new-user-name Robert
```

Cette commande ne produit aucune sortie.

Pour plus d'informations, consultez Affectation d'un nouveau nom à un groupe IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateUserà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.UpdateUserRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
```

```
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <curName> <newName>\s

                Where:
                    curName - The current user name.\s
                    newName - An updated user name.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String curName = args[0];
        String newName = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateIAMUser(iam, curName, newName);
        System.out.println("Done");
        iam.close();
    }

    public static void updateIAMUser(IamClient iam, String curName, String
 newName) {
        try {
            UpdateUserRequest request = UpdateUserRequest.builder()
                    .userName(curName)
                    .newUserName(newName)
                    .build();

            iam.updateUser(request);
            System.out.printf("Successfully updated user to username %s",
 newName);
```

```
        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, voir UpdateUserla section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Mettez à jour l'utilisateur.

```
import { UpdateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} currentUserName
 * @param {string} newUserName
 */
export const updateUser = (currentUserName, newUserName) => {
  const command = new UpdateUserCommand({
    UserName: currentUserName,
    NewUserName: newUserName,
  });

  return client.send(command);
};
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, voir UpdateUserla section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
  NewUserName: process.argv[3],
};

iam.updateUser(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK pour JavaScript.
- Pour plus de détails sur l'API, voir UpdateUserla section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun updateIAMUser(
    curName: String?,
    newName: String?,
) {
    val request =
        UpdateUserRequest {
            userName = curName
            newUserName = newName
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.updateUser(request)
        println("Successfully updated user to $newName")
    }
}
```

- Pour plus de détails sur l'API, consultez UpdateUserla section AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple renomme l'utilisateur IAM **`Bob`** en **`Robert`**.

```
Update-IAMUser -UserName Bob -NewUserName Robert
```

Exemple 2 : cet exemple modifie le chemin de l'utilisateur IAM **Bob** en **/Org1/Org2/**, ce qui change effectivement l'ARN de l'utilisateur en **arn:aws:iam::123456789012:user/Org1/Org2/bob**.

```
Update-IAMUser -UserName Bob -NewPath /Org1/Org2/
```

- Pour plus de détails sur l'API, reportez-vous UpdateUserà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple renomme l'utilisateur IAM **Bob** en **Robert**.

```
Update-IAMUser -UserName Bob -NewUserName Robert
```

Exemple 2 : cet exemple modifie le chemin de l'utilisateur IAM **Bob** en **/Org1/Org2/**, ce qui change effectivement l'ARN de l'utilisateur en **arn:aws:iam::123456789012:user/Org1/Org2/bob**.

```
Update-IAMUser -UserName Bob -NewPath /Org1/Org2/
```

- Pour plus de détails sur l'API, reportez-vous UpdateUserà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def update_user(user_name, new_user_name):
    """
    Updates a user's name.

    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
```

```
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user
```

- Pour plus de détails sur l'API, consultez UpdateUserle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# Updates an IAM user's name
#
# @param current_name [String] The current name of the user
# @param new_name [String] The new name of the user
def update_user_name(current_name, new_name)
  @iam_client.update_user(user_name: current_name, new_user_name: new_name)
  true
rescue StandardError => e
  @logger.error("Error updating user name from '#{current_name}' to
'#{new_name}': #{e.message}")
  false
end
```

- Pour plus de détails sur l'API, voir UpdateUserla section Référence des AWS SDK pour Ruby API.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    lo_iam->updateuser(
      iv_username = iv_user_name
      iv_newusername = iv_new_user_name ).
    MESSAGE 'User updated successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'User does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iamentityalrdyexex.
    MESSAGE 'New user name already exists.' TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, consultez UpdateUserla section de référence du AWS SDK pour l'API SAP ABAP.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`UploadServerCertificate`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `UploadServerCertificate`.

CLI

AWS CLI

Pour télécharger un certificat de serveur sur votre AWS compte

La commande upload-server-certificate suivante télécharge un certificat de serveur sur votre compte. AWS Dans cet exemple, le certificat se trouve dans le fichier `public_key_cert_file.pem`, la clé privée associée se trouve dans le fichier `my_private_key.pem` et la chaîne de certificats fournie par l'autorité de certification (CA) se trouve dans le fichier `my_certificate_chain_file.pem`. Lorsque le téléchargement du fichier est terminé, il est disponible sous le nom my ServerCertificate. Les paramètres commençant par `file://` indiquent à la commande de lire le contenu du fichier et de l'utiliser comme valeur de paramètre au lieu du nom du fichier lui-même.

```
aws iam upload-server-certificate \
    --server-certificate-name myServerCertificate \
    --certificate-body file://public_key_cert_file.pem \
    --private-key file://my_private_key.pem \
    --certificate-chain file://my_certificate_chain_file.pem
```

Sortie :

```
{
    "ServerCertificateMetadata": {
        "Path": "/",
        "ServerCertificateName": "myServerCertificate",
        "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
        "Arn": "arn:aws:iam::1234567989012:server-certificate/
myServerCertificate",
        "UploadDate": "2019-04-22T21:13:44+00:00",
        "Expiration": "2019-10-15T22:23:16+00:00"
    }
}
```

Pour plus d'informations, consultez Creating, Uploading, and Deleting Server Certificates dans le guide d'utilisation d'IAM.

- Pour plus de détails sur l'API, voir UploadServerCertificatela section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { UploadServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "node:fs";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";
import * as path from "node:path";

const client = new IAMClient({});

const certMessage = `Generate a certificate and key with the following command,
 or the equivalent for your system.

openssl req -x509 -newkey rsa:4096 -sha256 -days 3650 -nodes \
-keyout example.key -out example.crt -subj "/CN=example.com" \
-addext "subjectAltName=DNS:example.com,DNS:www.example.net,IP:10.0.0.1"
`;

const getCertAndKey = () => {
  try {
    const cert = readFileSync(
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.crt"),
    );
    const key = readFileSync(
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.key"),
    );
    return { cert, key };
  } catch (err) {
    if (err.code === "ENOENT") {
      throw new Error(
        `Certificate and/or private key not found. ${certMessage}`,
      );
    }

    throw err;
```

```
  }
};

/**
 *
 * @param {string} certificateName
 */
export const uploadServerCertificate = (certificateName) => {
  const { cert, key } = getCertAndKey();
  const command = new UploadServerCertificateCommand({
    ServerCertificateName: certificateName,
    CertificateBody: cert.toString(),
    PrivateKey: key.toString(),
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, voir UploadServerCertificatela section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : cet exemple charge un nouveau certificat de serveur sur le compte IAM. Les fichiers contenant le corps du certificat, la clé privée et (éventuellement) la chaîne de certificats doivent tous être codés au format PEM. Notez que les paramètres nécessitent le contenu réel des fichiers plutôt que les noms de fichiers. Vous devez utiliser le paramètre booléen **`-Raw`** pour traiter correctement le contenu du fichier.

```
Publish-IAMServerCertificate -ServerCertificateName MyTestCert -CertificateBody
 (Get-Content -Raw server.crt) -PrivateKey (Get-Content -Raw server.key)
```

Sortie :

```
Arn                   : arn:aws:iam::123456789012:server-certificate/MyTestCert
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /
ServerCertificateId   : ASCAJIEXAMPLE7J7HQZYW
```

```
ServerCertificateName : MyTestCert
UploadDate            : 4/21/2015 11:14:16 AM
```

- Pour plus de détails sur l'API, reportez-vous UploadServerCertificateà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : cet exemple charge un nouveau certificat de serveur sur le compte IAM. Les fichiers contenant le corps du certificat, la clé privée et (éventuellement) la chaîne de certificats doivent tous être codés au format PEM. Notez que les paramètres nécessitent le contenu réel des fichiers plutôt que les noms de fichiers. Vous devez utiliser le paramètre booléen **`-Raw`** pour traiter correctement le contenu du fichier.

```
Publish-IAMServerCertificate -ServerCertificateName MyTestCert -CertificateBody
 (Get-Content -Raw server.crt) -PrivateKey (Get-Content -Raw server.key)
```

Sortie :

```
Arn                 : arn:aws:iam::123456789012:server-certificate/MyTestCert
Expiration          : 1/14/2018 9:52:36 AM
Path                : /
ServerCertificateId : ASCAJIEXAMPLE7J7HQZYW
ServerCertificateName : MyTestCert
UploadDate          : 4/21/2015 11:14:16 AM
```

- Pour plus de détails sur l'API, reportez-vous UploadServerCertificateà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`UploadSigningCertificate`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `UploadSigningCertificate`.

CLI

AWS CLI

Pour charger un certificat de signature pour un utilisateur IAM

La commande `upload-signing-certificate` suivante charge un certificat de signature pour l'utilisateur IAM nommé Bob.

```
aws iam upload-signing-certificate \
    --user-name Bob \
    --certificate-body file://certificate.pem
```

Sortie :

```
{
    "Certificate": {
        "UserName": "Bob",
        "Status": "Active",
        "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-body>-----END
 CERTIFICATE-----",
        "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
        "UploadDate": "2013-06-06T21:40:08.121Z"
    }
}
```

Le certificat se trouve dans un fichier au format PEM nommé certificate.pem.

Pour plus d'informations, consultez Création, chargement et suppression de certificats de serveur dans le guide Utilisation d'IAM.

- Pour plus de détails sur l'API, reportez-vous UploadSigningCertificateà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple télécharge un nouveau certificat de X.509 signature et l'associe à l'utilisateur IAM nommé. **Bob** Le fichier contenant le corps du certificat est codé au format PEM. Le paramètre **CertificateBody** nécessite le contenu réel du fichier de certificat plutôt que le nom du fichier. Vous devez utiliser le paramètre booléen **-Raw** pour traiter correctement le fichier.

```
Publish-IAMSigningCertificate -UserName Bob -CertificateBody (Get-Content -Raw
 SampleSigningCert.pem)
```

Sortie :

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCEXAMPLEHMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Pour plus de détails sur l'API, reportez-vous UploadSigningCertificateà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple télécharge un nouveau certificat de X.509 signature et l'associe à l'utilisateur IAM nommé. **Bob** Le fichier contenant le corps du certificat est codé au format

PEM. Le paramètre **`CertificateBody`** nécessite le contenu réel du fichier de certificat plutôt que le nom du fichier. Vous devez utiliser le paramètre booléen **`-Raw`** pour traiter correctement le fichier.

```
Publish-IAMSigningCertificate -UserName Bob -CertificateBody (Get-Content -Raw
 SampleSigningCert.pem)
```

Sortie :

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCEXAMPLEHMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Pour plus de détails sur l'API, reportez-vous UploadSigningCertificateà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Scénarios d'utilisation d'IAM AWS Kits SDK

Les exemples de code suivants vous montrent comment implémenter des scénarios courants dans IAM avec des AWS SDK. Ces scénarios vous montrent comment accomplir des tâches spécifiques en appelant plusieurs fonctions dans IAM ou en les combinant avec d'autres Services AWS. Chaque exemple inclut un lien vers le code source complet, où vous trouverez des instructions sur la configuration et l'exécution du code.

Les scénarios ciblent un niveau d'expérience intermédiaire pour vous aider à comprendre les actions de service dans leur contexte.

Exemples

- Créez et gérez un service résilient à l'aide d'un AWS Kit SDK
- Configuration de la connectivité des services de conteneurs
- Création d'une tâche de conteneur pour le type de lancement sans serveur
- Création d'une API REST avec intégration de proxy de fonctions
- Créez des utilisateurs IAM en lecture seule et en lecture-écriture à l'aide d'un AWS Kit SDK
- Création d'un service de conteneur pour les instances de machines virtuelles
- Création d'un espace de travail de surveillance géré
- Création d'un tableau de bord de surveillance avec le nom de la fonction comme variable
- Création de votre première fonction sans serveur
- Commencez avec les entrepôts de données sans serveur à l'aide de la CLI
- Commencer à gérer la configuration
- Débuter avec la protection des appareils liés à l'Internet des objets
- Commencer à utiliser les magasins de fonctionnalités d'apprentissage automatique
- Commencer à utiliser les clusters Kubernetes gérés
- Débuter avec le streaming géré

- Commencer à utiliser des clusters d'entrepôts de données provisionnés
- Commencer à orchestrer les flux de travail
- Gérez les clés d'accès IAM à l'aide d'un AWS Kit SDK
- Gérez les politiques IAM à l'aide d'un AWS Kit SDK
- Gérez les rôles IAM à l'aide d'un AWS Kit SDK
- Gérez votre compte IAM à l'aide d'un AWS Kit SDK
- Transférer les secrets codés en dur vers un stockage secret sécurisé
- Permet le Optimiseur de calcul AWS Fonction d'automatisation pour appliquer les actions recommandées
- Politique pour activer l'automatisation dans toute votre organisation
- Politique pour activer l'automatisation pour votre compte
- Politique pour accorder un accès complet à l'automatisation de l'optimiseur de calcul au compte de gestion d'une organisation
- Politique visant à accorder un accès complet à Compute Optimizer Automation en mode autonome AWS comptes
- Politique pour accorder un accès en lecture seule à l'automatisation de l'optimiseur de calcul au compte de gestion d'une organisation
- Politique visant à accorder un accès en lecture seule à Compute Optimizer Automation en mode autonome AWS comptes
- Politique pour accorder les autorisations de rôle lié à un service pour l'automatisation de l'optimisation du calcul
- Restaurer une version d'une politique IAM à l'aide d'un AWS Kit SDK
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts
- Configurer le contrôle Attribute-Based d'accès pour DynamoDB à l'aide de AWS Command Line Interface v2
- Configuration de la gestion des systèmes
- Utilisation de variables de propriété dans les tableaux de bord de surveillance pour surveiller plusieurs fonctions sans serveur
- Travaillez avec DynamoDB Streams et utilisez Time-to-Live AWS Command Line Interface v2
- Utilisez l'API IAM Policy Builder à l'aide d'un AWS Kit SDK

Créez et gérez un service résilient à l'aide d'un AWS Kit SDK

Les exemples de code suivant montrent comment créer un service web à charge équilibrée qui renvoie des recommandations de livres, de films et de chansons. L'exemple montre comment le service répond aux défaillances et comment le restructurer pour accroître la résilience en cas de défaillance.

- Utilisez un groupe Amazon EC2 Auto Scaling pour créer des instances Amazon Elastic Compute Cloud (Amazon EC2) sur la base d'un modèle de lancement et pour maintenir le nombre d'instances dans une plage spécifiée.
- Gérez et distribuez les requêtes HTTP avec Elastic Load Balancing.
- Surveillez l'état des instances d'un groupe Auto Scaling et transférez les demandes uniquement aux instances saines.
- Exécutez un serveur Web Python sur chaque instance EC2 pour gérer les requêtes HTTP. Le serveur Web répond par des recommandations et des surveillances de l'état.
- Simulez un service de recommandation avec une table Amazon DynamoDB.
- Contrôlez la réponse du serveur Web aux demandes et aux contrôles de santé en mettant à jour AWS Systems Manager les paramètres.

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Exécutez un scénario interactif à une invite de commande.

```
    static async Task Main(string[] args)
    {
        _configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally, load local settings.
```

```
            .Build();


        // Set up dependency injection for the AWS services.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonIdentityManagementService>()
                    .AddAWSService<IAmazonDynamoDB>()
                    .AddAWSService<IAmazonElasticLoadBalancingV2>()
                    .AddAWSService<IAmazonSimpleSystemsManagement>()
                    .AddAWSService<IAmazonAutoScaling>()
                    .AddAWSService<IAmazonEC2>()
                    .AddTransient<AutoScalerWrapper>()
                    .AddTransient<ElasticLoadBalancerWrapper>()
                    .AddTransient<SmParameterWrapper>()
                    .AddTransient<Recommendations>()
                    .AddSingleton<IConfiguration>(_configuration)
            )
            .Build();

        ServicesSetup(host);
        ResourcesSetup();

        try
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Welcome to the Resilient Architecture Example
Scenario.");
            Console.WriteLine(new string('-', 80));
            await Deploy(true);

            Console.WriteLine("Now let's begin the scenario.");
            Console.WriteLine(new string('-', 80));
            await Demo(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Finally, let's clean up our resources.");
            Console.WriteLine(new string('-', 80));
```

```
            await DestroyResources(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Resilient Architecture Example Scenario is
complete.");
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine($"There was a problem running the scenario:
{ex.Message}");
            await DestroyResources(true);
            Console.WriteLine(new string('-', 80));
        }
    }

    /// <summary>
    /// Setup any common resources, also used for integration testing.
    /// </summary>
    public static void ResourcesSetup()
    {
        _httpClient = new HttpClient();
    }

    /// <summary>
    /// Populate the services for use within the console application.
    /// </summary>
    /// <param name="host">The services host.</param>
    private static void ServicesSetup(IHost host)
    {
        _elasticLoadBalancerWrapper =
host.Services.GetRequiredService<ElasticLoadBalancerWrapper>();
        _iamClient =
host.Services.GetRequiredService<IAmazonIdentityManagementService>();
        _recommendations = host.Services.GetRequiredService<Recommendations>();
        _autoScalerWrapper =
host.Services.GetRequiredService<AutoScalerWrapper>();
        _smParameterWrapper =
host.Services.GetRequiredService<SmParameterWrapper>();
    }

    /// <summary>
```

```
    /// Deploy necessary resources for the scenario.
    /// </summary>
    /// <param name="interactive">True to run as interactive.</param>
    /// <returns>True if successful.</returns>
    public static async Task<bool> Deploy(bool interactive)
    {
        var protocol = "HTTP";
        var port = 80;
        var sshPort = 22;

        Console.WriteLine(
            "\nFor this demo, we'll use the AWS SDK for .NET to create several
AWS resources\n" +
            "to set up a load-balanced web service endpoint and explore some ways
to make it resilient\n" +
            "against various kinds of failures.\n\n" +
            "Some of the resources create by this demo are:\n");

        Console.WriteLine(
            "\t* A DynamoDB table that the web service depends on to provide
book, movie, and song recommendations.");
        Console.WriteLine(
            "\t* An EC2 launch template that defines EC2 instances that each
contain a Python web server.");
        Console.WriteLine(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
several Availability Zones.");
        Console.WriteLine(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
Auto Scaling group to distribute requests.");
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to start deploying
resources.");
        if (interactive)
            Console.ReadLine();

        // Create and populate the DynamoDB table.
        var databaseTableName = _configuration["databaseName"];
        var recommendationsPath = Path.Join(_configuration["resourcePath"],
            "recommendations_objects.json");
        Console.WriteLine($"Creating and populating a DynamoDB table named
{databaseTableName}.");
        await _recommendations.CreateDatabaseWithName(databaseTableName);
```

```
        await _recommendations.PopulateDatabase(databaseTableName,
recommendationsPath);
        Console.WriteLine(new string('-', 80));

        // Create the EC2 Launch Template.

        Console.WriteLine(
            $"Creating an EC2 launch template that runs
'server_startup_script.sh' when an instance starts.\n"
            + "\nThis script starts a Python web server defined in the
`server.py` script. The web server\n"
            + "listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.\n"
            + "For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
            + "run a web server, such as Apache, with least-privileged
credentials.");
        Console.WriteLine(
            "\nThe template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            + "permissions to access the DynamoDB recommendation table and
Systems Manager parameters\n"
            + "that control the flow of the demo.");

        var startupScriptPath = Path.Join(_configuration["resourcePath"],
            "server_startup_script.sh");
        var instancePolicyPath = Path.Join(_configuration["resourcePath"],
            "instance_policy.json");
        await _autoScalerWrapper.CreateTemplate(startupScriptPath,
instancePolicyPath);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            + "Availability Zone.\n");
        var zones = await _autoScalerWrapper.DescribeAvailabilityZones();
        await _autoScalerWrapper.CreateGroupOfSize(3,
_autoScalerWrapper.GroupName, zones);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
```

```
            + "HTTP requests. You can see these instances in the console or
continue with the demo.\n");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("Creating variables that control the flow of the
demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine(
            "\nCreating an Elastic Load Balancing target group and load balancer.
The target group\n"
            + "defines how the load balancer connects to instances. The load
balancer provides a\n"
            + "single endpoint where clients connect and dispatches requests to
instances in the group.");

        var defaultVpc = await _autoScalerWrapper.GetDefaultVpc();
        var subnets = await
_autoScalerWrapper.GetAllVpcSubnetsForZones(defaultVpc.VpcId, zones);
        var subnetIds = subnets.Select(s => s.SubnetId).ToList();
        var targetGroup = await
_elasticLoadBalancerWrapper.CreateTargetGroupOnVpc(_elasticLoadBalancerWrapper.TargetGro
protocol, port, defaultVpc.VpcId);

        await
_elasticLoadBalancerWrapper.CreateLoadBalancerAndListener(_elasticLoadBalancerWrapper.Lo
subnetIds, targetGroup);
        await
_autoScalerWrapper.AttachLoadBalancerToGroup(_autoScalerWrapper.GroupName,
targetGroup.TargetGroupArn);
        Console.WriteLine("\nVerifying access to the load balancer endpoint...");
        var endPoint = await
_elasticLoadBalancerWrapper.GetEndpointForLoadBalancerByName(_elasticLoadBalancerWrapper
        var loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);

        if (!loadBalancerAccess)
        {
            Console.WriteLine("\nCouldn't connect to the load balancer, verifying
that the port is open...");
```

```
            var ipString = await _httpClient.GetStringAsync("https://
checkip.amazonaws.com");
            ipString = ipString.Trim();

            var defaultSecurityGroup = await
 _autoScalerWrapper.GetDefaultSecurityGroupForVpc(defaultVpc);
            var portIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, port,
 ipString);
            var sshPortIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, sshPort,
 ipString);

            if (!portIsOpen)
            {
                Console.WriteLine(
                    "\nFor this example to work, the default security group for
 your default VPC must\n"
                    + "allows access from this computer. You can either add it
 automatically from this\n"
                    + "example or add it yourself using the AWS Management
 Console.\n");

                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
 inbound traffic from your computer's IP address?"))
                {
                    await
 _autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, port,
 ipString);
                }
            }

            if (!sshPortIsOpen)
            {
                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
 inbound SSH traffic for debugging from your computer's IP address?"))
                {
                    await
 _autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, sshPort,
 ipString);
                }
```

```
            }
            loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);
        }

        if (loadBalancerAccess)
        {
            Console.WriteLine("Your load balancer is ready. You can access it by
browsing to:");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        else
        {
            Console.WriteLine(
                "\nCouldn't get a successful response from the load balancer
endpoint. Troubleshoot by\n"
                + "manually verifying that your VPC and security group are
configured correctly and that\n"
                + "you can successfully make a GET request to the load balancer
endpoint:\n");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue with the
demo.");
        if (interactive)
            Console.ReadLine();
        return true;
    }

    /// <summary>
    /// Demonstrate the steps of the scenario.
    /// </summary>
    /// <param name="interactive">True to run as an interactive scenario.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> Demo(bool interactive)
    {
        var ssmOnlyPolicy = Path.Join(_configuration["resourcePath"],
            "ssm_only_policy.json");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Resetting parameters to starting values for demo.");
        await _smParameterWrapper.Reset();
```

```
        Console.WriteLine("\nThis part of the demonstration shows how to toggle
different parts of the system\n" +
                          "to create situations where the web service fails, and
shows how using a resilient\n" +
                          "architecture can keep the web service running in spite
of these failures.");
        Console.WriteLine(new string('-', 88));
        Console.WriteLine("At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine($"The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.\n" +
                          $"The table name is contained in a Systems Manager
parameter named '{_smParameterWrapper.TableParameter}'.\n" +
                          $"To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");
        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as\n" +
                          "healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Instead of failing when the recommendation service
fails, the web service can return a static response.");
        Console.WriteLine("While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.FailureResponseParameter,
"static");

        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a static response.");
        Console.WriteLine("The service still reports as healthy because health
checks are still shallow.");
        if (interactive)
            await DemoActionChoices();
```

```
        Console.WriteLine("Let's reinstate the recommendation service.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
_smParameterWrapper.TableName);
        Console.WriteLine(
            "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n" +
            "access the DynamoDB recommendation table.\n"
        );
        await _autoScalerWrapper.CreateInstanceProfileWithName(
            _autoScalerWrapper.BadCredsPolicyName,
            _autoScalerWrapper.BadCredsRoleName,
            _autoScalerWrapper.BadCredsProfileName,
            ssmOnlyPolicy,
            new List<string> { "AmazonSSMManagedInstanceCore" }
        );
        var instances = await
_autoScalerWrapper.GetInstancesByGroupName(_autoScalerWrapper.GroupName);
        var badInstanceId = instances.First();
        var instanceProfile = await
_autoScalerWrapper.GetInstanceProfile(badInstanceId);
        Console.WriteLine(
            $"Replacing the profile for instance {badInstanceId} with a profile
that contains\n" +
            "bad credentials...\n"
        );
        await _autoScalerWrapper.ReplaceInstanceProfile(
            badInstanceId,
            _autoScalerWrapper.BadCredsProfileName,
            instanceProfile.AssociationId
        );
        Console.WriteLine(
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n" +
            "depending on which instance is selected by the load balancer.\n"
        );
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nLet's implement a deep health check. For this demo,
a deep health check tests whether");
        Console.WriteLine("the web service can access the DynamoDB table that it
depends on for recommendations. Note that");
```

```
        Console.WriteLine("the deep health check is only for ELB routing and not
for Auto Scaling instance health.");
        Console.WriteLine("This kind of deep health check is not recommended for
Auto Scaling instance health, because it");
        Console.WriteLine("risks accidental termination of all instances in the
Auto Scaling group when a dependent service fails.");

        Console.WriteLine("\nBy implementing deep health checks, the load
balancer can detect when one of the instances is failing");
        Console.WriteLine("and take that instance out of rotation.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.HealthCheckParameter,
"deep");

        Console.WriteLine($"\nNow, checking target health indicates that the
instance with bad credentials ({badInstanceId})");
        Console.WriteLine("is unhealthy. Note that it might take a minute or two
for the load balancer to detect the unhealthy");
        Console.WriteLine("instance. Sending a GET request to the load balancer
endpoint always returns a recommendation, because");
        Console.WriteLine("the load balancer takes unhealthy instances out of its
rotation.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nBecause the instances in this demo are controlled by
an auto scaler, the simplest way to fix an unhealthy");
        Console.WriteLine("instance is to terminate it and let the auto scaler
start a new instance to replace it.");

        await _autoScalerWrapper.TryTerminateInstanceById(badInstanceId);

        Console.WriteLine($"\nEven while the instance is terminating and the new
instance is starting, sending a GET");
        Console.WriteLine("request to the web service continues to get a
successful recommendation response because");
        Console.WriteLine("starts and reports as healthy, it is included in the
load balancing rotation.");
        Console.WriteLine("Note that terminating and replacing an instance
typically takes several minutes, during which time you");
        Console.WriteLine("can see the changing health check status until the new
instance is running and healthy.");
```

```
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nIf the recommendation service fails now, deep health
checks mean all instances report as unhealthy.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");

        Console.WriteLine($"\nWhen all instances are unhealthy, the load balancer
continues to route requests even to");
        Console.WriteLine("unhealthy instances, allowing them to fail open and
return a static response rather than fail");
        Console.WriteLine("closed and report failure to the customer.");

        if (interactive)
            await DemoActionChoices();
        await _smParameterWrapper.Reset();

        Console.WriteLine(new string('-', 80));
        return true;
    }

    /// <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <param name="interactive">True to ask the user for cleanup.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> DestroyResources(bool interactive)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine(
            "To keep things tidy and to avoid unwanted charges on your account,
we can clean up all AWS resources\n" +
            "that were created for this demo."
        );

        if (!interactive || GetYesNoResponse("Do you want to clean up all demo
resources? (y/n) "))
        {
            await
_elasticLoadBalancerWrapper.DeleteLoadBalancerByName(_elasticLoadBalancerWrapper.LoadBal
```

```
            await
_elasticLoadBalancerWrapper.DeleteTargetGroupByName(_elasticLoadBalancerWrapper.TargetGr
            await
_autoScalerWrapper.TerminateAndDeleteAutoScalingGroupWithName(_autoScalerWrapper.GroupNa
            await
_autoScalerWrapper.DeleteKeyPairByName(_autoScalerWrapper.KeyPairName);
            await
_autoScalerWrapper.DeleteTemplateByName(_autoScalerWrapper.LaunchTemplateName);
            await _autoScalerWrapper.DeleteInstanceProfile(
                _autoScalerWrapper.BadCredsProfileName,
                _autoScalerWrapper.BadCredsRoleName
            );
            await
_recommendations.DestroyDatabaseByName(_recommendations.TableName);
        }
        else
        {
            Console.WriteLine(
                "Ok, we'll leave the resources intact.\n" +
                "Don't forget to delete them when you're done with them or you
might incur unexpected charges."
            );
        }

        Console.WriteLine(new string('-', 80));
        return true;
    }
```

Créez une classe qui englobe les actions Auto Scaling et Amazon EC2.

```
/// <summary>
/// Encapsulates Amazon EC2 Auto Scaling and EC2 management methods.
/// </summary>
public class AutoScalerWrapper
{
    private readonly IAmazonAutoScaling _amazonAutoScaling;
    private readonly IAmazonEC2 _amazonEc2;
    private readonly IAmazonSimpleSystemsManagement _amazonSsm;
    private readonly IAmazonIdentityManagementService _amazonIam;
    private readonly ILogger<AutoScalerWrapper> _logger;

    private readonly string _instanceType = "";
```

```
    private readonly string _amiParam = "";
    private readonly string _launchTemplateName = "";
    private readonly string _groupName = "";
    private readonly string _instancePolicyName = "";
    private readonly string _instanceRoleName = "";
    private readonly string _instanceProfileName = "";
    private readonly string _badCredsProfileName = "";
    private readonly string _badCredsRoleName = "";
    private readonly string _badCredsPolicyName = "";
    private readonly string _keyPairName = "";

    public string GroupName => _groupName;
    public string KeyPairName => _keyPairName;
    public string LaunchTemplateName => _launchTemplateName;
    public string InstancePolicyName => _instancePolicyName;
    public string BadCredsProfileName => _badCredsProfileName;
    public string BadCredsRoleName => _badCredsRoleName;
    public string BadCredsPolicyName => _badCredsPolicyName;

    /// <summary>
    /// Constructor for the AutoScalerWrapper.
    /// </summary>
    /// <param name="amazonAutoScaling">The injected AutoScaling client.</param>
    /// <param name="amazonEc2">The injected EC2 client.</param>
    /// <param name="amazonIam">The injected IAM client.</param>
    /// <param name="amazonSsm">The injected SSM client.</param>
    public AutoScalerWrapper(
        IAmazonAutoScaling amazonAutoScaling,
        IAmazonEC2 amazonEc2,
        IAmazonSimpleSystemsManagement amazonSsm,
        IAmazonIdentityManagementService amazonIam,
        IConfiguration configuration,
        ILogger<AutoScalerWrapper> logger)
    {
        _amazonAutoScaling = amazonAutoScaling;
        _amazonEc2 = amazonEc2;
        _amazonSsm = amazonSsm;
        _amazonIam = amazonIam;
        _logger = logger;

        var prefix = configuration["resourcePrefix"];
        _instanceType = configuration["instanceType"];
        _amiParam = configuration["amiParam"];
```

```
        _launchTemplateName = prefix + "-template";
        _groupName = prefix + "-group";
        _instancePolicyName = prefix + "-pol";
        _instanceRoleName = prefix + "-role";
        _instanceProfileName = prefix + "-prof";
        _badCredsPolicyName = prefix + "-bc-pol";
        _badCredsRoleName = prefix + "-bc-role";
        _badCredsProfileName = prefix + "-bc-prof";
        _keyPairName = prefix + "-key-pair";
    }

    /// <summary>
    /// Create a policy, role, and profile that is associated with instances with
a specified name.
    /// An instance's associated profile defines a role that is assumed by the
    /// instance.The role has attached policies that specify the AWS permissions
granted to
    /// clients that run on the instance.
    /// </summary>
    /// <param name="policyName">Name to use for the policy.</param>
    /// <param name="roleName">Name to use for the role.</param>
    /// <param name="profileName">Name to use for the profile.</param>
    /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
    /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
    /// <returns>The Arn of the profile.</returns>
    public async Task<string> CreateInstanceProfileWithName(
        string policyName,
        string roleName,
        string profileName,
        string ssmOnlyPolicyFile,
        List<string>? awsManagedPolicies = null)
    {

        var assumeRoleDoc = "{" +
                                    "\"Version\": \"2012-10-17\"," +
                                    "\"Statement\": [{" +
                                         "\"Effect\": \"Allow\"," +
                                         "\"Principal\": {" +
                                         "\"Service\": [" +
                                             "\"ec2.amazonaws.com\"" +
                                         "]" +
                                         "}," +
                                    "\"Action\": \"sts:AssumeRole\"" +
```

```
                                "}]" +
                            "}";

        var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

        var policyArn = "";

        try
        {
            var createPolicyResult = await _amazonIam.CreatePolicyAsync(
                new CreatePolicyRequest
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
```

```
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
```

```
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }

    /// <summary>
    /// Create a new key pair and save the file.
    /// </summary>
    /// <param name="newKeyPairName">The name of the new key pair.</param>
    /// <returns>Async task.</returns>
    public async Task CreateKeyPair(string newKeyPairName)
    {
        try
        {
            var keyResponse = await _amazonEc2.CreateKeyPairAsync(
                new CreateKeyPairRequest() { KeyName = newKeyPairName });
            await File.WriteAllTextAsync($"{newKeyPairName}.pem",
                keyResponse.KeyPair.KeyMaterial);
            Console.WriteLine($"Created key pair {newKeyPairName}.");
        }
        catch (AlreadyExistsException)
        {
            Console.WriteLine("Key pair already exists.");
        }
    }

    /// <summary>
    /// Delete the key pair and file by name.
    /// </summary>
    /// <param name="deleteKeyPairName">The key pair to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteKeyPairByName(string deleteKeyPairName)
    {
        try
```

```
        {
            await _amazonEc2.DeleteKeyPairAsync(
                new DeleteKeyPairRequest() { KeyName = deleteKeyPairName });
            File.Delete($"{deleteKeyPairName}.pem");
        }
        catch (FileNotFoundException)
        {
            Console.WriteLine($"Key pair {deleteKeyPairName} not found.");
        }
    }

    /// <summary>
    /// Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling.
    /// The launch template specifies a Bash script in its user data field that
runs after
    /// the instance is started. This script installs the Python packages and
starts a Python
    /// web server on the instance.
    /// </summary>
    /// <param name="startupScriptPath">The path to a Bash script file that is
run.</param>
    /// <param name="instancePolicyPath">The path to a permissions policy to
create and attach to the profile.</param>
    /// <returns>The template object.</returns>
    public async Task<Amazon.EC2.Model.LaunchTemplate> CreateTemplate(string
startupScriptPath, string instancePolicyPath)
    {
        try
        {
            await CreateKeyPair(_keyPairName);
            await CreateInstanceProfileWithName(_instancePolicyName,
_instanceRoleName,
                _instanceProfileName, instancePolicyPath);

            var startServerText = await File.ReadAllTextAsync(startupScriptPath);
            var plainTextBytes =
System.Text.Encoding.UTF8.GetBytes(startServerText);

            var amiLatest = await _amazonSsm.GetParameterAsync(
                new GetParameterRequest() { Name = _amiParam });
            var amiId = amiLatest.Parameter.Value;
            var launchTemplateResponse = await
_amazonEc2.CreateLaunchTemplateAsync(
```

```
                new CreateLaunchTemplateRequest()
                {
                    LaunchTemplateName = _launchTemplateName,
                    LaunchTemplateData = new RequestLaunchTemplateData()
                    {
                        InstanceType = _instanceType,
                        ImageId = amiId,
                        IamInstanceProfile =
                            new

LaunchTemplateIamInstanceProfileSpecificationRequest()
                            {
                                Name = _instanceProfileName
                            },
                        KeyName = _keyPairName,
                        UserData = System.Convert.ToBase64String(plainTextBytes)
                    }
                });
            return launchTemplateResponse.LaunchTemplate;
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode ==
"InvalidLaunchTemplateName.AlreadyExistsException")
            {
                _logger.LogError($"Could not create the template, the name
{_launchTemplateName} already exists. " +
                                 $"Please try again with a unique name.");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while creating the template.:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Get a list of Availability Zones in the AWS Region of the Amazon EC2
Client.
    /// </summary>
```

```
    /// <returns>A list of availability zones.</returns>
    public async Task<List<string>> DescribeAvailabilityZones()
    {
        try
        {
            var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
                new DescribeAvailabilityZonesRequest());
            return zoneResponse.AvailabilityZones.Select(z =>
 z.ZoneName).ToList();
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            _logger.LogError($"An Amazon EC2 error occurred while listing
 availability zones.: {ec2Exception.Message}");
            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while listing availability
 zones.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Create an EC2 Auto Scaling group of a specified size and name.
    /// </summary>
    /// <param name="groupSize">The size for the group.</param>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="availabilityZones">The availability zones for the group.</
param>
    /// <returns>Async task.</returns>
    public async Task CreateGroupOfSize(int groupSize, string groupName,
 List<string> availabilityZones)
    {
        try
        {
            await _amazonAutoScaling.CreateAutoScalingGroupAsync(
                new CreateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    AvailabilityZones = availabilityZones,
                    LaunchTemplate =
```

```
                        new
Amazon.AutoScaling.Model.LaunchTemplateSpecification()
                        {
                            LaunchTemplateName = _launchTemplateName,
                            Version = "$Default"
                        },
                    MaxSize = groupSize,
                    MinSize = groupSize
                });
            Console.WriteLine($"Created EC2 Auto Scaling group {groupName} with
size {groupSize}.");
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine($"EC2 Auto Scaling group {groupName} already
exists.");
        }
    }

    /// <summary>
    /// Get the default VPC for the account.
    /// </summary>
    /// <returns>The default VPC object.</returns>
    public async Task<Vpc> GetDefaultVpc()
    {
        try
        {
            var vpcResponse = await _amazonEc2.DescribeVpcsAsync(
                new DescribeVpcsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("is-default", new List<string>() { "true" })
                    }
                });
            return vpcResponse.Vpcs[0];
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "UnauthorizedOperation")
            {
                _logger.LogError(ec2Exception, $"You do not have the necessary
permissions to describe VPCs.");
            }
```

```
            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while describing the vpcs.:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Get all the subnets for a Vpc in a set of availability zones.
    /// </summary>
    /// <param name="vpcId">The Id of the Vpc.</param>
    /// <param name="availabilityZones">The list of availability zones.</param>
    /// <returns>The collection of subnet objects.</returns>
    public async Task<List<Subnet>> GetAllVpcSubnetsForZones(string vpcId,
List<string> availabilityZones)
    {
        try
        {
            var subnets = new List<Subnet>();
            var subnetPaginator = _amazonEc2.Paginators.DescribeSubnets(
                new DescribeSubnetsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("vpc-id", new List<string>() { vpcId }),
                        new("availability-zone", availabilityZones),
                        new("default-for-az", new List<string>() { "true" })
                    }
                });

            // Get the entire list using the paginator.
            await foreach (var subnet in subnetPaginator.Subnets)
            {
                subnets.Add(subnet);
            }

            return subnets;
        }
        catch (AmazonEC2Exception ec2Exception)
        {
```

```
            if (ec2Exception.ErrorCode == "InvalidVpcID.NotFound")
            {
                _logger.LogError(ec2Exception, $"The specified VPC ID {vpcId}
does not exist.");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while describing the
subnets.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Delete a launch template by name.
    /// </summary>
    /// <param name="templateName">The name of the template to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTemplateByName(string templateName)
    {
        try
        {
            await _amazonEc2.DeleteLaunchTemplateAsync(
                new DeleteLaunchTemplateRequest()
                {
                    LaunchTemplateName = templateName
                });
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode ==
"InvalidLaunchTemplateName.NotFoundException")
            {
                _logger.LogError(
                    $"Could not delete the template, the name
{_launchTemplateName} was not found.");
            }

            throw;
        }
        catch (Exception ex)
```

```
        {
            _logger.LogError($"An error occurred while deleting the template.:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Detaches a role from an instance profile, detaches policies from the
role,
    /// and deletes all the resources.
    /// </summary>
    /// <param name="profileName">The name of the profile to delete.</param>
    /// <param name="roleName">The name of the role to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteInstanceProfile(string profileName, string roleName)
    {
        try
        {
            await _amazonIam.RemoveRoleFromInstanceProfileAsync(
                new RemoveRoleFromInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
            await _amazonIam.DeleteInstanceProfileAsync(
                new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
            var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
```

```
                        {
                            PolicyArn = policy.PolicyArn
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
    }

    /// <summary>
    /// Gets data about the instances in an EC2 Auto Scaling group by its group
name.
    /// </summary>
    /// <param name="group">The name of the auto scaling group.</param>
    /// <returns>A collection of instance Ids.</returns>
    public async Task<IEnumerable<string>> GetInstancesByGroupName(string group)
    {
        var instanceResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { group }
            });
        var instanceIds = instanceResponse.AutoScalingGroups.SelectMany(
            g => g.Instances.Select(i => i.InstanceId));
        return instanceIds;
    }

    /// <summary>
    /// Get the instance profile association data for an instance.
    /// </summary>
    /// <param name="instanceId">The Id of the instance.</param>
    /// <returns>Instance profile associations data.</returns>
    public async Task<IamInstanceProfileAssociation> GetInstanceProfile(string
instanceId)
    {
        try
        {
```

```
            var response = await
_amazonEc2.DescribeIamInstanceProfileAssociationsAsync(
                new DescribeIamInstanceProfileAssociationsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("instance-id", new List<string>() { instanceId })
                    },
                });
            return response.IamInstanceProfileAssociations[0];
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
            {
                _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while creating the
template.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Replace the profile associated with a running instance. After the profile
is replaced, the instance
    /// is rebooted to ensure that it uses the new profile. When the instance is
ready, Systems Manager is
    /// used to restart the Python web server.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to update.</param>
    /// <param name="credsProfileName">The name of the new profile to associate
with the specified instance.</param>
    /// <param name="associationId">The Id of the existing profile association
for the instance.</param>
    /// <returns>Async task.</returns>
    public async Task ReplaceInstanceProfile(string instanceId, string
credsProfileName, string associationId)
```

```
    {
        try
        {
            await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
                new ReplaceIamInstanceProfileAssociationRequest()
                {
                    AssociationId = associationId,
                    IamInstanceProfile = new IamInstanceProfileSpecification()
                    {
                        Name = credsProfileName
                    }
                });
            // Allow time before resetting.
            Thread.Sleep(25000);

            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(25000);
            var instanceReady = false;
            var retries = 5;
            while (retries-- > 0 && !instanceReady)
            {
                var instancesPaginator =
                    _amazonSsm.Paginators.DescribeInstanceInformation(
                        new DescribeInstanceInformationRequest());
                // Get the entire list using the paginator.
                await foreach (var instance in
instancesPaginator.InstanceInformationList)
                {
                    instanceReady = instance.InstanceId == instanceId;
                    if (instanceReady)
                    {
                        break;
                    }
                }
            }
            Console.WriteLine("Waiting for instance to be running.");
            await WaitForInstanceState(instanceId, InstanceStateName.Running);
            Console.WriteLine("Instance ready.");
            Console.WriteLine($"Sending restart command to instance
{instanceId}");
            await _amazonSsm.SendCommandAsync(
                new SendCommandRequest()
                {
```

```
                    InstanceIds = new List<string>() { instanceId },
                    DocumentName = "AWS-RunShellScript",
                    Parameters = new Dictionary<string, List<string>>()
                    {
                        {
                            "commands",
                            new List<string>() { "cd / && sudo python3 server.py
80" }
                        }
                    }
                });
            Console.WriteLine($"Restarted the web server on instance
{instanceId}");
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
            {
                _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while replacing the
template.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Try to terminate an instance by its Id.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task TryTerminateInstanceById(string instanceId)
    {
        var stopping = false;
        Console.WriteLine($"Stopping {instanceId}...");
        while (!stopping)
        {
            try
```

```
            {
                await
_amazonAutoScaling.TerminateInstanceInAutoScalingGroupAsync(
                    new TerminateInstanceInAutoScalingGroupRequest()
                    {
                        InstanceId = instanceId,
                        ShouldDecrementDesiredCapacity = false
                    });
                stopping = true;
            }
            catch (ScalingActivityInProgressException)
            {
                Console.WriteLine($"Scaling activity in progress for
{instanceId}. Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
    /// waits and retries until the group is successfully deleted.
    /// </summary>
    /// <param name="groupName">The name of the group to try to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TryDeleteGroupByName(string groupName)
    {
        var stopped = false;
        while (!stopped)
        {
            try
            {
                await _amazonAutoScaling.DeleteAutoScalingGroupAsync(
                    new DeleteAutoScalingGroupRequest()
                    {
                        AutoScalingGroupName = groupName
                    });
                stopped = true;
            }
            catch (Exception e)
                when ((e is ScalingActivityInProgressException)
                      || (e is Amazon.AutoScaling.Model.ResourceInUseException))
            {
```

```
                Console.WriteLine($"Some instances are still running.
Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Terminate instances and delete the Auto Scaling group by name.
    /// </summary>
    /// <param name="groupName">The name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TerminateAndDeleteAutoScalingGroupWithName(string
groupName)
    {
        var describeGroupsResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { groupName }
            });
        if (describeGroupsResponse.AutoScalingGroups.Any())
        {
            // Update the size to 0.
            await _amazonAutoScaling.UpdateAutoScalingGroupAsync(
                new UpdateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    MinSize = 0
                });
            var group = describeGroupsResponse.AutoScalingGroups[0];
            foreach (var instance in group.Instances)
            {
                await TryTerminateInstanceById(instance.InstanceId);
            }

            await TryDeleteGroupByName(groupName);
        }
        else
        {
            Console.WriteLine($"No groups found with name {groupName}.");
        }
    }
```

```
    /// <summary>
    /// Get the default security group for a specified Vpc.
    /// </summary>
    /// <param name="vpc">The Vpc to search.</param>
    /// <returns>The default security group.</returns>
    public async Task<SecurityGroup> GetDefaultSecurityGroupForVpc(Vpc vpc)
    {
        var groupResponse = await _amazonEc2.DescribeSecurityGroupsAsync(
            new DescribeSecurityGroupsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("group-name", new List<string>() { "default" }),
                    new ("vpc-id", new List<string>() { vpc.VpcId })
                }
            });
        return groupResponse.SecurityGroups[0];
    }

    /// <summary>
    /// Verify the default security group of a Vpc allows ingress from the
calling computer.
    /// This can be done by allowing ingress from this computer's IP address.
    /// In some situations, such as connecting from a corporate network, you must
instead specify
    /// a prefix list Id. You can also temporarily open the port to any IP
address while running this example.
    /// If you do, be sure to remove public access when you're done.
    /// </summary>
    /// <param name="vpc">The group to check.</param>
    /// <param name="port">The port to verify.</param>
    /// <param name="ipAddress">This computer's IP address.</param>
    /// <returns>True if the ip address is allowed on the group.</returns>
    public bool VerifyInboundPortForGroup(SecurityGroup group, int port, string
ipAddress)
    {
        var portIsOpen = false;
        foreach (var ipPermission in group.IpPermissions)
        {
            if (ipPermission.FromPort == port)
            {
                foreach (var ipRange in ipPermission.Ipv4Ranges)
                {
```

```
                    var cidr = ipRange.CidrIp;
                    if (cidr.StartsWith(ipAddress) || cidr == "0.0.0.0/0")
                    {
                        portIsOpen = true;
                    }
                }

                if (ipPermission.PrefixListIds.Any())
                {
                    portIsOpen = true;
                }

                if (!portIsOpen)
                {
                    Console.WriteLine("The inbound rule does not appear to be
open to either this computer's IP\n" +
                                      "address, to all IP addresses (0.0.0.0/0),
or to a prefix list ID.");
                }
                else
                {
                    break;
                }
            }
        }

        return portIsOpen;
    }

    /// <summary>
    /// Add an ingress rule to the specified security group that allows access on
the
    /// specified port from the specified IP address.
    /// </summary>
    /// <param name="groupId">The Id of the security group to modify.</param>
    /// <param name="port">The port to open.</param>
    /// <param name="ipAddress">The IP address to allow access.</param>
    /// <returns>Async task.</returns>
    public async Task OpenInboundPort(string groupId, int port, string ipAddress)
    {
        await _amazonEc2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest()
            {
                GroupId = groupId,
```

```
                IpPermissions = new List<IpPermission>()
                {
                    new IpPermission()
                    {
                        FromPort = port,
                        ToPort = port,
                        IpProtocol = "tcp",
                        Ipv4Ranges = new List<IpRange>()
                        {
                            new IpRange() { CidrIp = $"{ipAddress}/32" }
                        }
                    }
                }
            });
    }

    /// <summary>
    /// Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
 Scaling group.
    /// The
    /// </summary>
    /// <param name="autoScalingGroupName">The name of the Auto Scaling group.</
param>
    /// <param name="targetGroupArn">The Arn for the target group.</param>
    /// <returns>Async task.</returns>
    public async Task AttachLoadBalancerToGroup(string autoScalingGroupName,
 string targetGroupArn)
    {
        await _amazonAutoScaling.AttachLoadBalancerTargetGroupsAsync(
            new AttachLoadBalancerTargetGroupsRequest()
            {
                AutoScalingGroupName = autoScalingGroupName,
                TargetGroupARNs = new List<string>() { targetGroupArn }
            });
    }

    /// <summary>
    /// Wait until an EC2 instance is in a specified state.
    /// </summary>
    /// <param name="instanceId">The instance Id.</param>
    /// <param name="stateName">The state to wait for.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitForInstanceState(string instanceId,
 InstanceStateName stateName)
```

```
    {
        var request = new DescribeInstancesRequest
        {
            InstanceIds = new List<string> { instanceId }
        };

        // Wait until the instance is in the specified state.
        var hasState = false;
        do
        {
            // Wait 5 seconds.
            Thread.Sleep(5000);

            // Check for the desired state.
            var response = await _amazonEc2.DescribeInstancesAsync(request);
            var instance = response.Reservations[0].Instances[0];
            hasState = instance.State.Name == stateName;
            Console.Write(". ");
        } while (!hasState);

        return hasState;
    }
}
```

Créez une classe qui englobe les actions Elastic Load Balancing.

```
/// <summary>
/// Encapsulates Elastic Load Balancer actions.
/// </summary>
public class ElasticLoadBalancerWrapper
{
    private readonly IAmazonElasticLoadBalancingV2 _amazonElasticLoadBalancingV2;
    private string? _endpoint = null;
    private readonly string _targetGroupName = "";
    private readonly string _loadBalancerName = "";
    HttpClient _httpClient = new();

    public string TargetGroupName => _targetGroupName;
    public string LoadBalancerName => _loadBalancerName;

    /// <summary>
```

```
    /// Constructor for the Elastic Load Balancer wrapper.
    /// </summary>
    /// <param name="amazonElasticLoadBalancingV2">The injected load balancing v2
client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public ElasticLoadBalancerWrapper(
        IAmazonElasticLoadBalancingV2 amazonElasticLoadBalancingV2,
        IConfiguration configuration)
    {
        _amazonElasticLoadBalancingV2 = amazonElasticLoadBalancingV2;
        var prefix = configuration["resourcePrefix"];
        _targetGroupName = prefix + "-tg";
        _loadBalancerName = prefix + "-lb";
    }

    /// <summary>
    /// Get the HTTP Endpoint of a load balancer by its name.
    /// </summary>
    /// <param name="loadBalancerName">The name of the load balancer.</param>
    /// <returns>The HTTP endpoint.</returns>
    public async Task<string> GetEndpointForLoadBalancerByName(string
loadBalancerName)
    {
        if (_endpoint == null)
        {
            var endpointResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { loadBalancerName }
                    });
            _endpoint = endpointResponse.LoadBalancers[0].DNSName;
        }

        return _endpoint;
    }

    /// <summary>
    /// Return the GET response for an endpoint as text.
    /// </summary>
    /// <param name="endpoint">The endpoint for the request.</param>
    /// <returns>The request response.</returns>
    public async Task<string> GetEndPointResponse(string endpoint)
    {
```

```
        var endpointResponse = await _httpClient.GetAsync($"http://{endpoint}");
        var textResponse = await endpointResponse.Content.ReadAsStringAsync();
        return textResponse!;
    }

    /// <summary>
    /// Get the target health for a group by name.
    /// </summary>
    /// <param name="groupName">The name of the group.</param>
    /// <returns>The collection of health descriptions.</returns>
    public async Task<List<TargetHealthDescription>>
CheckTargetHealthForGroup(string groupName)
    {
        List<TargetHealthDescription> result = null!;
        try
        {
            var groupResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                    new DescribeTargetGroupsRequest()
                    {
                        Names = new List<string>() { groupName }
                    });
            var healthResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetHealthAsync(
                    new DescribeTargetHealthRequest()
                    {
                        TargetGroupArn =
groupResponse.TargetGroups[0].TargetGroupArn
                    });
            ;
            result = healthResponse.TargetHealthDescriptions;
        }
        catch (TargetGroupNotFoundException)
        {
            Console.WriteLine($"Target group {groupName} not found.");
        }
        return result;
    }

    /// <summary>
    /// Create an Elastic Load Balancing target group. The target group specifies
how the load balancer forwards
    /// requests to instances in the group and how instance health is checked.
    ///
```

```
    /// To speed up this demo, the health check is configured with shortened
 times and lower thresholds. In production,
    /// you might want to decrease the sensitivity of your health checks to avoid
 unwanted failures.
    /// </summary>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="protocol">The protocol, such as HTTP.</param>
    /// <param name="port">The port to use to forward requests, such as 80.</
param>
    /// <param name="vpcId">The Id of the Vpc in which the load balancer
 exists.</param>
    /// <returns>The new TargetGroup object.</returns>
    public async Task<TargetGroup> CreateTargetGroupOnVpc(string groupName,
 ProtocolEnum protocol, int port, string vpcId)
    {
        var createResponse = await
 _amazonElasticLoadBalancingV2.CreateTargetGroupAsync(
            new CreateTargetGroupRequest()
            {
                Name = groupName,
                Protocol = protocol,
                Port = port,
                HealthCheckPath = "/healthcheck",
                HealthCheckIntervalSeconds = 10,
                HealthCheckTimeoutSeconds = 5,
                HealthyThresholdCount = 2,
                UnhealthyThresholdCount = 2,
                VpcId = vpcId
            });
        var targetGroup = createResponse.TargetGroups[0];
        return targetGroup;
    }

    /// <summary>
    /// Create an Elastic Load Balancing load balancer that uses the specified
 subnets
    /// and forwards requests to the specified target group.
    /// </summary>
    /// <param name="name">The name for the new load balancer.</param>
    /// <param name="subnetIds">Subnets for the load balancer.</param>
    /// <param name="targetGroup">Target group for forwarded requests.</param>
    /// <returns>The new LoadBalancer object.</returns>
    public async Task<LoadBalancer> CreateLoadBalancerAndListener(string name,
 List<string> subnetIds, TargetGroup targetGroup)
```

```
    {
        var createLbResponse = await
_amazonElasticLoadBalancingV2.CreateLoadBalancerAsync(
            new CreateLoadBalancerRequest()
            {
                Name = name,
                Subnets = subnetIds
            });
        var loadBalancerArn = createLbResponse.LoadBalancers[0].LoadBalancerArn;

        // Wait for load balancer to be available.
        var loadBalancerReady = false;
        while (!loadBalancerReady)
        {
            try
            {
                var describeResponse =
                    await
_amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                        new DescribeLoadBalancersRequest()
                        {
                            Names = new List<string>() { name }
                        });

                var loadBalancerState =
describeResponse.LoadBalancers[0].State.Code;

                loadBalancerReady = loadBalancerState ==
LoadBalancerStateEnum.Active;
            }
            catch (LoadBalancerNotFoundException)
            {
                loadBalancerReady = false;
            }
            Thread.Sleep(10000);
        }
        // Create the listener.
        await _amazonElasticLoadBalancingV2.CreateListenerAsync(
            new CreateListenerRequest()
            {
                LoadBalancerArn = loadBalancerArn,
                Protocol = targetGroup.Protocol,
                Port = targetGroup.Port,
                DefaultActions = new List<Action>()
```

```
                {
                    new Action()
                    {
                        Type = ActionTypeEnum.Forward,
                        TargetGroupArn = targetGroup.TargetGroupArn
                    }
                }
            });
        return createLbResponse.LoadBalancers[0];
    }

    /// <summary>
    /// Verify this computer can successfully send a GET request to the
    /// load balancer endpoint.
    /// </summary>
    /// <param name="endpoint">The endpoint to check.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> VerifyLoadBalancerEndpoint(string endpoint)
    {
        var success = false;
        var retries = 3;
        while (!success && retries > 0)
        {
            try
            {
                var endpointResponse = await _httpClient.GetAsync($"http://
{endpoint}");
                Console.WriteLine($"Response: {endpointResponse.StatusCode}.");

                if (endpointResponse.IsSuccessStatusCode)
                {
                    success = true;
                }
                else
                {
                    retries = 0;
                }
            }
            catch (HttpRequestException)
            {
                Console.WriteLine("Connection error, retrying...");
                retries--;
                Thread.Sleep(10000);
            }
```

```
        }

        return success;
    }

    /// <summary>
    /// Delete a load balancer by its specified name.
    /// </summary>
    /// <param name="name">The name of the load balancer to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteLoadBalancerByName(string name)
    {
        try
        {
            var describeLoadBalancerResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { name }
                    });
            var lbArn =
describeLoadBalancerResponse.LoadBalancers[0].LoadBalancerArn;
            await _amazonElasticLoadBalancingV2.DeleteLoadBalancerAsync(
                new DeleteLoadBalancerRequest()
                {
                    LoadBalancerArn = lbArn
                }
            );
        }
        catch (LoadBalancerNotFoundException)
        {
            Console.WriteLine($"Load balancer {name} not found.");
        }
    }

    /// <summary>
    /// Delete a TargetGroup by its specified name.
    /// </summary>
    /// <param name="groupName">Name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTargetGroupByName(string groupName)
    {
        var done = false;
        while (!done)
```

```
        {
            try
            {
                var groupResponse =
                    await
 _amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                        new DescribeTargetGroupsRequest()
                        {
                            Names = new List<string>() { groupName }
                        });

                var targetArn = groupResponse.TargetGroups[0].TargetGroupArn;
                await _amazonElasticLoadBalancingV2.DeleteTargetGroupAsync(
                    new DeleteTargetGroupRequest() { TargetGroupArn =
 targetArn });
                Console.WriteLine($"Deleted load balancing target group
 {groupName}.");
                done = true;
            }
            catch (TargetGroupNotFoundException)
            {
                Console.WriteLine(
                    $"Target group {groupName} not found, could not delete.");
                done = true;
            }
            catch (ResourceInUseException)
            {
                Console.WriteLine("Target group not yet released, waiting...");
                Thread.Sleep(10000);
            }
        }
    }
}
```

Créez une classe qui utilise DynamoDB pour simuler un service de recommandation.

```
/// <summary>
/// Encapsulates a DynamoDB table to use as a service that recommends books,
 movies, and songs.
/// </summary>
public class Recommendations
{
```

```
    private readonly IAmazonDynamoDB _amazonDynamoDb;
    private readonly DynamoDBContext _context;
    private readonly string _tableName;

    public string TableName => _tableName;

    /// <summary>
    /// Constructor for the Recommendations service.
    /// </summary>
    /// <param name="amazonDynamoDb">The injected DynamoDb client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public Recommendations(IAmazonDynamoDB amazonDynamoDb, IConfiguration
configuration)
    {
        _amazonDynamoDb = amazonDynamoDb;
        _context = new DynamoDBContext(_amazonDynamoDb);
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Create the DynamoDb table with a specified name.
    /// </summary>
    /// <param name="tableName">The name for the table.</param>
    /// <returns>True when ready.</returns>
    public async Task<bool> CreateDatabaseWithName(string tableName)
    {
        try
        {
            Console.Write($"Creating table {tableName}...");
            var createRequest = new CreateTableRequest()
            {
                TableName = tableName,
                AttributeDefinitions = new List<AttributeDefinition>()
                    {
                        new AttributeDefinition()
                        {
                            AttributeName = "MediaType",
                            AttributeType = ScalarAttributeType.S
                        },
                        new AttributeDefinition()
                        {
                            AttributeName = "ItemId",
                            AttributeType = ScalarAttributeType.N
                        }
```

```
                    },
                KeySchema = new List<KeySchemaElement>()
                    {
                        new KeySchemaElement()
                        {
                            AttributeName = "MediaType",
                            KeyType = KeyType.HASH
                        },
                        new KeySchemaElement()
                        {
                            AttributeName = "ItemId",
                            KeyType = KeyType.RANGE
                        }
                    },
                ProvisionedThroughput = new ProvisionedThroughput()
                {
                    ReadCapacityUnits = 5,
                    WriteCapacityUnits = 5
                }
            };
            await _amazonDynamoDb.CreateTableAsync(createRequest);

            // Wait until the table is ACTIVE and then report success.
            Console.Write("\nWaiting for table to become active...");

            var request = new DescribeTableRequest
            {
                TableName = tableName
            };

            TableStatus status;
            do
            {
                Thread.Sleep(2000);

                var describeTableResponse = await
_amazonDynamoDb.DescribeTableAsync(request);
                status = describeTableResponse.Table.TableStatus;

                Console.Write(".");
            }
            while (status != "ACTIVE");

            return status == TableStatus.ACTIVE;
```

```
        }
        catch (ResourceInUseException)
        {
            Console.WriteLine($"Table {tableName} already exists.");
            return false;
        }
    }

    /// <summary>
    /// Populate the database table with data from a specified path.
    /// </summary>
    /// <param name="databaseTableName">The name of the table.</param>
    /// <param name="recommendationsPath">The path of the recommendations data.</
param>
    /// <returns>Async task.</returns>
    public async Task PopulateDatabase(string databaseTableName, string
 recommendationsPath)
    {
        var recommendationsText = await
 File.ReadAllTextAsync(recommendationsPath);
        var records =

 JsonSerializer.Deserialize<RecommendationModel[]>(recommendationsText);
        var batchWrite = _context.CreateBatchWrite<RecommendationModel>();

        foreach (var record in records!)
        {
            batchWrite.AddPutItem(record);
        }

        await batchWrite.ExecuteAsync();
    }

    /// <summary>
    /// Delete the recommendation table by name.
    /// </summary>
    /// <param name="tableName">The name of the recommendation table.</param>
    /// <returns>Async task.</returns>
    public async Task DestroyDatabaseByName(string tableName)
    {
        try
        {
            await _amazonDynamoDb.DeleteTableAsync(
                new DeleteTableRequest() { TableName = tableName });
```

```
            Console.WriteLine($"Table {tableName} was deleted.");
        }
        catch (ResourceNotFoundException)
        {
            Console.WriteLine($"Table {tableName} not found");
        }
    }
}
```

Créez une classe qui englobe les actions de Systems Manager.

```
/// <summary>
/// Encapsulates Systems Manager parameter operations. This example uses these
 parameters
/// to drive the demonstration of resilient architecture, such as failure of a
 dependency or
/// how the service responds to a health check.
/// </summary>
public class SmParameterWrapper
{
    private readonly IAmazonSimpleSystemsManagement
 _amazonSimpleSystemsManagement;

    private readonly string _tableParameter = "doc-example-resilient-
architecture-table";
    private readonly string _failureResponseParameter = "doc-example-resilient-
architecture-failure-response";
    private readonly string _healthCheckParameter = "doc-example-resilient-
architecture-health-check";
    private readonly string _tableName = "";

    public string TableParameter => _tableParameter;
    public string TableName => _tableName;
    public string HealthCheckParameter => _healthCheckParameter;
    public string FailureResponseParameter => _failureResponseParameter;

    /// <summary>
    /// Constructor for the SmParameterWrapper.
    /// </summary>
    /// <param name="amazonSimpleSystemsManagement">The injected Simple Systems
 Management client.</param>
    /// <param name="configuration">The injected configuration.</param>
```

```
    public SmParameterWrapper(IAmazonSimpleSystemsManagement
 amazonSimpleSystemsManagement, IConfiguration configuration)
    {
        _amazonSimpleSystemsManagement = amazonSimpleSystemsManagement;
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Reset the Systems Manager parameters to starting values for the demo.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task Reset()
    {
        await this.PutParameterByName(_tableParameter, _tableName);
        await this.PutParameterByName(_failureResponseParameter, "none");
        await this.PutParameterByName(_healthCheckParameter, "shallow");
    }

    /// <summary>
    /// Set the value of a named Systems Manager parameter.
    /// </summary>
    /// <param name="name">The name of the parameter.</param>
    /// <param name="value">The value to set.</param>
    /// <returns>Async task.</returns>
    public async Task PutParameterByName(string name, string value)
    {
        await _amazonSimpleSystemsManagement.PutParameterAsync(
            new PutParameterRequest() { Name = name, Value = value, Overwrite =
 true });
    }
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la référence de l'API AWS SDK pour .NET .
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer

- CreateTargetGroup
- DeleteAutoScalingGroup
- DeleteInstanceProfile
- DeleteLaunchTemplate
- DeleteLoadBalancer
- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Exécutez un scénario interactif à une invite de commande.

```
public class Main {
```

```
    public static final String fileName = "C:\\AWS\\resworkflow\
\recommendations.json"; // Modify file location.
    public static final String tableName = "doc-example-recommendation-service";
    public static final String startScript = "C:\\AWS\\resworkflow\
\server_startup_script.sh"; // Modify file location.
    public static final String policyFile = "C:\\AWS\\resworkflow\
\instance_policy.json"; // Modify file location.
    public static final String ssmJSON = "C:\\AWS\\resworkflow\
\ssm_only_policy.json"; // Modify file location.
    public static final String failureResponse = "doc-example-resilient-
architecture-failure-response";
    public static final String healthCheck = "doc-example-resilient-architecture-
health-check";
    public static final String templateName = "doc-example-resilience-template";
    public static final String roleName = "doc-example-resilience-role";
    public static final String policyName = "doc-example-resilience-pol";
    public static final String profileName = "doc-example-resilience-prof";

    public static final String badCredsProfileName = "doc-example-resilience-
prof-bc";

    public static final String targetGroupName = "doc-example-resilience-tg";
    public static final String autoScalingGroupName = "doc-example-resilience-
group";
    public static final String lbName = "doc-example-resilience-lb";
    public static final String protocol = "HTTP";
    public static final int port = 80;

    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws IOException,
 InterruptedException {
        Scanner in = new Scanner(System.in);
        Database database = new Database();
        AutoScaler autoScaler = new AutoScaler();
        LoadBalancer loadBalancer = new LoadBalancer();

        System.out.println(DASHES);
        System.out.println("Welcome to the demonstration of How to Build and
 Manage a Resilient Service!");
        System.out.println(DASHES);

        System.out.println(DASHES);
```

```
        System.out.println("A - SETUP THE RESOURCES");
        System.out.println("Press Enter when you're ready to start deploying
resources.");
        in.nextLine();
        deploy(loadBalancer);
        System.out.println(DASHES);
        System.out.println(DASHES);
        System.out.println("B - DEMO THE RESILIENCE FUNCTIONALITY");
        System.out.println("Press Enter when you're ready.");
        in.nextLine();
        demo(loadBalancer);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("C - DELETE THE RESOURCES");
        System.out.println("""
                This concludes the demo of how to build and manage a resilient
service.
                To keep things tidy and to avoid unwanted charges on your
account, we can clean up all AWS resources
                that were created for this demo.
                """);

        System.out.println("\n Do you want to delete the resources (y/n)? ");
        String userInput = in.nextLine().trim().toLowerCase(); // Capture user
input

        if (userInput.equals("y")) {
            // Delete resources here
            deleteResources(loadBalancer, autoScaler, database);
            System.out.println("Resources deleted.");
        } else {
            System.out.println("""
                    Okay, we'll leave the resources intact.
                    Don't forget to delete them when you're done with them or you
might incur unexpected charges.
                    """);
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("The example has completed. ");
        System.out.println("\n Thanks for watching!");
        System.out.println(DASHES);
```

```
   }

   // Deletes the AWS resources used in this example.
   private static void deleteResources(LoadBalancer loadBalancer, AutoScaler
autoScaler, Database database)
           throws IOException, InterruptedException {
       loadBalancer.deleteLoadBalancer(lbName);
       System.out.println("*** Wait 30 secs for resource to be deleted");
       TimeUnit.SECONDS.sleep(30);
       loadBalancer.deleteTargetGroup(targetGroupName);
       autoScaler.deleteAutoScaleGroup(autoScalingGroupName);
       autoScaler.deleteRolesPolicies(policyName, roleName, profileName);
       autoScaler.deleteTemplate(templateName);
       database.deleteTable(tableName);
   }

   private static void deploy(LoadBalancer loadBalancer) throws
InterruptedException, IOException {
       Scanner in = new Scanner(System.in);
       System.out.println(
               """
                       For this demo, we'll use the AWS SDK for Java (v2) to
create several AWS resources
                       to set up a load-balanced web service endpoint and
explore some ways to make it resilient
                       against various kinds of failures.

                       Some of the resources create by this demo are:
                       \t* A DynamoDB table that the web service depends on to
provide book, movie, and song recommendations.
                       \t* An EC2 launch template that defines EC2 instances
that each contain a Python web server.
                       \t* An EC2 Auto Scaling group that manages EC2 instances
across several Availability Zones.
                       \t* An Elastic Load Balancing (ELB) load balancer that
targets the Auto Scaling group to distribute requests.
                       """);

       System.out.println("Press Enter when you're ready.");
       in.nextLine();
       System.out.println(DASHES);

       System.out.println(DASHES);
```

```
        System.out.println("Creating and populating a DynamoDB table named " +
tableName);
        Database database = new Database();
        database.createTable(tableName, fileName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.
                This script starts a Python web server defined in the `server.py`
script. The web server
                listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.
                For demo purposes, this server is run as the root user. In
production, the best practice is to
                run a web server, such as Apache, with least-privileged
credentials.

                The template also defines an IAM policy that each instance uses
to assume a role that grants
                permissions to access the DynamoDB recommendation table and
Systems Manager parameters
                that control the flow of the demo.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        templateCreator.createTemplate(policyFile, policyName, profileName,
startScript, templateName, roleName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(
                "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different Availability Zone.");
        System.out.println("*** Wait 30 secs for the VPC to be created");
        TimeUnit.SECONDS.sleep(30);
        AutoScaler autoScaler = new AutoScaler();
        String[] zones = autoScaler.createGroup(3, templateName,
autoScalingGroupName);

        System.out.println("""
                At this point, you have EC2 instances created. Once each instance
starts, it listens for
```

```
                HTTP requests. You can see these instances in the console or
continue with the demo.
                Press Enter when you're ready to continue.
                """);

        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Creating variables that control the flow of the
demo.");
        ParameterHelper paramHelper = new ParameterHelper();
        paramHelper.reset();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an Elastic Load Balancing target group and load
balancer. The target group
                defines how the load balancer connects to instances. The load
balancer provides a
                single endpoint where clients connect and dispatches requests to
instances in the group.
                """);

        String vpcId = autoScaler.getDefaultVPC();
        List<Subnet> subnets = autoScaler.getSubnets(vpcId, zones);
        System.out.println("You have retrieved a list with " + subnets.size() + "
subnets");
        String targetGroupArn = loadBalancer.createTargetGroup(protocol, port,
vpcId, targetGroupName);
        String elbDnsName = loadBalancer.createLoadBalancer(subnets,
targetGroupArn, lbName, port, protocol);
        autoScaler.attachLoadBalancerTargetGroup(autoScalingGroupName,
targetGroupArn);
        System.out.println("Verifying access to the load balancer endpoint...");
        boolean wasSuccessul =
loadBalancer.verifyLoadBalancerEndpoint(elbDnsName);
        if (!wasSuccessul) {
            System.out.println("Couldn't connect to the load balancer, verifying
that the port is open...");
            CloseableHttpClient httpClient = HttpClients.createDefault();

            // Create an HTTP GET request to "http://checkip.amazonaws.com"
```

```
            HttpGet httpGet = new HttpGet("http://checkip.amazonaws.com");
            try {
                // Execute the request and get the response
                HttpResponse response = httpClient.execute(httpGet);

                // Read the response content.
                String ipAddress =
IOUtils.toString(response.getEntity().getContent(),
StandardCharsets.UTF_8).trim();

                // Print the public IP address.
                System.out.println("Public IP Address: " + ipAddress);
                GroupInfo groupInfo = autoScaler.verifyInboundPort(vpcId, port,
ipAddress);
                if (!groupInfo.isPortOpen()) {
                    System.out.println("""
                            For this example to work, the default security group
for your default VPC must
                            allow access from this computer. You can either add
it automatically from this
                            example or add it yourself using the AWS Management
Console.
                            """);

                    System.out.println(
                            "Do you want to add a rule to security group " +
groupInfo.getGroupName() + " to allow");
                    System.out.println("inbound traffic on port " + port + " from
your computer's IP address (y/n) ");
                    String ans = in.nextLine();
                    if ("y".equalsIgnoreCase(ans)) {
                        autoScaler.openInboundPort(groupInfo.getGroupName(),
String.valueOf(port), ipAddress);
                        System.out.println("Security group rule added.");
                    } else {
                        System.out.println("No security group rule added.");
                    }
                }

            } catch (AutoScalingException e) {
                e.printStackTrace();
            }
        } else if (wasSuccessul) {
```

```
            System.out.println("Your load balancer is ready. You can access it by
browsing to:");
            System.out.println("\t http://" + elbDnsName);
        } else {
            System.out.println("Couldn't get a successful response from the load
balancer endpoint. Troubleshoot by");
            System.out.println("manually verifying that your VPC and security
group are configured correctly and that");
            System.out.println("you can successfully make a GET request to the
load balancer.");
        }

        System.out.println("Press Enter when you're ready to continue with the
demo.");
        in.nextLine();
    }

    // A method that controls the demo part of the Java program.
    public static void demo(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        ParameterHelper paramHelper = new ParameterHelper();
        System.out.println("Read the ssm_only_policy.json file");
        String ssmOnlyPolicy = readFileAsString(ssmJSON);

        System.out.println("Resetting parameters to starting values for demo.");
        paramHelper.reset();

        System.out.println(
                """
                        This part of the demonstration shows how to toggle
different parts of the system
                        to create situations where the web service fails, and
shows how using a resilient
                        architecture can keep the web service running in spite
of these failures.

                        At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.
                       """);
        demoChoices(loadBalancer);

        System.out.println(
                """
```

```
                        The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.
                        The table name is contained in a Systems Manager
parameter named self.param_helper.table.
                        To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.
                       """);
       paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

       System.out.println(
               """
                        \nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as
                        healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.
                       """);
       demoChoices(loadBalancer);

       System.out.println(
               """
                       Instead of failing when the recommendation service fails,
the web service can return a static response.
                       While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.
                       """);
       paramHelper.put(paramHelper.failureResponse, "static");

       System.out.println("""
               Now, sending a GET request to the load balancer endpoint returns
a static response.
               The service still reports as healthy because health checks are
still shallow.
               """);
       demoChoices(loadBalancer);

       System.out.println("Let's reinstate the recommendation service.");
       paramHelper.put(paramHelper.tableName, paramHelper.dyntable);

       System.out.println("""
               Let's also substitute bad credentials for one of the instances in
the target group so that it can't
               access the DynamoDB recommendation table. We will get an instance
id value.
               """);
```

```
        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        AutoScaler autoScaler = new AutoScaler();

        // Create a new instance profile based on badCredsProfileName.
        templateCreator.createInstanceProfile(policyFile, policyName,
badCredsProfileName, roleName);
        String badInstanceId = autoScaler.getBadInstance(autoScalingGroupName);
        System.out.println("The bad instance id values used for this demo is " +
badInstanceId);

        String profileAssociationId =
autoScaler.getInstanceProfile(badInstanceId);
        System.out.println("The association Id value is " +
profileAssociationId);
        System.out.println("Replacing the profile for instance " + badInstanceId
                + " with a profile that contains bad credentials");
        autoScaler.replaceInstanceProfile(badInstanceId, badCredsProfileName,
profileAssociationId);

        System.out.println(
                """
                        Now, sending a GET request to the load balancer endpoint
returns either a recommendation or a static response,
                        depending on which instance is selected by the load
balancer.
                        """);

        demoChoices(loadBalancer);

        System.out.println("""
                Let's implement a deep health check. For this demo, a deep health
check tests whether
                the web service can access the DynamoDB table that it depends on
for recommendations. Note that
                the deep health check is only for ELB routing and not for Auto
Scaling instance health.
                This kind of deep health check is not recommended for Auto
Scaling instance health, because it
                risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.
                """);

        System.out.println("""
```

```
                By implementing deep health checks, the load balancer can detect
when one of the instances is failing
                and take that instance out of rotation.
                """);

        paramHelper.put(paramHelper.healthCheck, "deep");

        System.out.println("""
                Now, checking target health indicates that the instance with bad
credentials
                is unhealthy. Note that it might take a minute or two for the
load balancer to detect the unhealthy
                instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because
                the load balancer takes unhealthy instances out of its rotation.
                """);

        demoChoices(loadBalancer);

        System.out.println(
                """
                        Because the instances in this demo are controlled by an
auto scaler, the simplest way to fix an unhealthy
                        instance is to terminate it and let the auto scaler start
a new instance to replace it.
                        """);
        autoScaler.terminateInstance(badInstanceId);

        System.out.println("""
                Even while the instance is terminating and the new instance is
starting, sending a GET
                request to the web service continues to get a successful
recommendation response because
                the load balancer routes requests to the healthy instances. After
the replacement instance
                starts and reports as healthy, it is included in the load
balancing rotation.
                Note that terminating and replacing an instance typically takes
several minutes, during which time you
                can see the changing health check status until the new instance
is running and healthy.
                """);

        demoChoices(loadBalancer);
```

```
        System.out.println(
                "If the recommendation service fails now, deep health checks mean
all instances report as unhealthy.");
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        demoChoices(loadBalancer);
        paramHelper.reset();
    }

    public static void demoChoices(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        String[] actions = {
                "Send a GET request to the load balancer endpoint.",
                "Check the health of load balancer targets.",
                "Go to the next part of the demo."
        };
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("-".repeat(88));
            System.out.println("See the current state of the service by selecting
one of the following choices:");
            for (int i = 0; i < actions.length; i++) {
                System.out.println(i + ": " + actions[i]);
            }

            try {
                System.out.print("\nWhich action would you like to take? ");
                int choice = scanner.nextInt();
                System.out.println("-".repeat(88));

                switch (choice) {
                    case 0 -> {
                        System.out.println("Request:\n");
                        System.out.println("GET http://" +
loadBalancer.getEndpoint(lbName));
                        CloseableHttpClient httpClient =
HttpClients.createDefault();

                        // Create an HTTP GET request to the ELB.
                        HttpGet httpGet = new HttpGet("http://" +
loadBalancer.getEndpoint(lbName));

                        // Execute the request and get the response.
```

```
                        HttpResponse response = httpClient.execute(httpGet);
                        int statusCode =
response.getStatusLine().getStatusCode();
                        System.out.println("HTTP Status Code: " + statusCode);

                        // Display the JSON response
                        BufferedReader reader = new BufferedReader(
                                new
InputStreamReader(response.getEntity().getContent()));
                        StringBuilder jsonResponse = new StringBuilder();
                        String line;
                        while ((line = reader.readLine()) != null) {
                            jsonResponse.append(line);
                        }
                        reader.close();

                        // Print the formatted JSON response.
                        System.out.println("Full Response:\n");
                        System.out.println(jsonResponse.toString());

                        // Close the HTTP client.
                        httpClient.close();

                    }
                    case 1 -> {
                        System.out.println("\nChecking the health of load
balancer targets:\n");
                        List<TargetHealthDescription> health =
loadBalancer.checkTargetHealth(targetGroupName);
                        for (TargetHealthDescription target : health) {
                            System.out.printf("\tTarget %s on port %d is %s%n",
target.target().id(),
                                    target.target().port(),
target.targetHealth().stateAsString());
                        }
                        System.out.println("""
                                Note that it can take a minute or two for the
health check to update
                                after changes are made.
                                """);
                    }
                    case 2 -> {
                        System.out.println("\nOkay, let's move on.");
                        System.out.println("-".repeat(88));
```

```
                        return; // Exit the method when choice is 2
                    }
                    default -> System.out.println("You must choose a value
 between 0-2. Please select again.");
                }

            } catch (java.util.InputMismatchException e) {
                System.out.println("Invalid input. Please select again.");
                scanner.nextLine(); // Clear the input buffer.
            }
        }
    }

    public static String readFileAsString(String filePath) throws IOException {
        byte[] bytes = Files.readAllBytes(Paths.get(filePath));
        return new String(bytes);
    }
}
```

Créez une classe qui englobe les actions Auto Scaling et Amazon EC2.

```
public class AutoScaler {

    private static Ec2Client ec2Client;
    private static AutoScalingClient autoScalingClient;
    private static IamClient iamClient;

    private static SsmClient ssmClient;

    private IamClient getIAMClient() {
        if (iamClient == null) {
            iamClient = IamClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return iamClient;
    }

    private SsmClient getSSMClient() {
        if (ssmClient == null) {
            ssmClient = SsmClient.builder()
                    .region(Region.US_EAST_1)
```

```
                    .build();
        }
        return ssmClient;
    }

    private Ec2Client getEc2Client() {
        if (ec2Client == null) {
            ec2Client = Ec2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ec2Client;
    }

    private AutoScalingClient getAutoScalingClient() {
        if (autoScalingClient == null) {
            autoScalingClient = AutoScalingClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return autoScalingClient;
    }

    /**
     * Terminates and instances in an EC2 Auto Scaling group. After an instance
is
     * terminated, it can no longer be accessed.
     */
    public void terminateInstance(String instanceId) {
        TerminateInstanceInAutoScalingGroupRequest terminateInstanceIRequest =
TerminateInstanceInAutoScalingGroupRequest
                .builder()
                .instanceId(instanceId)
                .shouldDecrementDesiredCapacity(false)
                .build();


getAutoScalingClient().terminateInstanceInAutoScalingGroup(terminateInstanceIRequest);
        System.out.format("Terminated instance %s.", instanceId);
    }

    /**
     * Replaces the profile associated with a running instance. After the profile
is
```

```
     * replaced, the instance is rebooted to ensure that it uses the new profile.
     * When
     * the instance is ready, Systems Manager is used to restart the Python web
     * server.
     */
    public void replaceInstanceProfile(String instanceId, String
newInstanceProfileName, String profileAssociationId)
            throws InterruptedException {
        // Create an IAM instance profile specification.
        software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
iamInstanceProfile =
software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
                .builder()
                .name(newInstanceProfileName) // Make sure
'newInstanceProfileName' is a valid IAM Instance Profile
                                              // name.
                .build();

        // Replace the IAM instance profile association for the EC2 instance.
        ReplaceIamInstanceProfileAssociationRequest replaceRequest =
ReplaceIamInstanceProfileAssociationRequest
                .builder()
                .iamInstanceProfile(iamInstanceProfile)
                .associationId(profileAssociationId) // Make sure
'profileAssociationId' is a valid association ID.
                .build();

        try {
            getEc2Client().replaceIamInstanceProfileAssociation(replaceRequest);
            // Handle the response as needed.
        } catch (Ec2Exception e) {
            // Handle exceptions, log, or report the error.
            System.err.println("Error: " + e.getMessage());
        }
        System.out.format("Replaced instance profile for association %s with
profile %s.", profileAssociationId,
                newInstanceProfileName);
        TimeUnit.SECONDS.sleep(15);
        boolean instReady = false;
        int tries = 0;

        // Reboot after 60 seconds
        while (!instReady) {
            if (tries % 6 == 0) {
```

```
                getEc2Client().rebootInstances(RebootInstancesRequest.builder()
                        .instanceIds(instanceId)
                        .build());
                System.out.println("Rebooting instance " + instanceId + " and
waiting for it to be ready.");
            }
            tries++;
            try {
                TimeUnit.SECONDS.sleep(10);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }

            DescribeInstanceInformationResponse informationResponse =
getSSMClient().describeInstanceInformation();
            List<InstanceInformation> instanceInformationList =
informationResponse.instanceInformationList();
            for (InstanceInformation info : instanceInformationList) {
                if (info.instanceId().equals(instanceId)) {
                    instReady = true;
                    break;
                }
            }
        }

        SendCommandRequest sendCommandRequest = SendCommandRequest.builder()
                .instanceIds(instanceId)
                .documentName("AWS-RunShellScript")
                .parameters(Collections.singletonMap("commands",
                        Collections.singletonList("cd / && sudo python3 server.py
80")))
                .build();

        getSSMClient().sendCommand(sendCommandRequest);
        System.out.println("Restarted the Python web server on instance " +
instanceId + ".");
    }

    public void openInboundPort(String secGroupId, String port, String ipAddress)
{
        AuthorizeSecurityGroupIngressRequest ingressRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                .groupName(secGroupId)
                .cidrIp(ipAddress)
```

```
                .fromPort(Integer.parseInt(port))
                .build();

        getEc2Client().authorizeSecurityGroupIngress(ingressRequest);
        System.out.format("Authorized ingress to %s on port %s from %s.",
secGroupId, port, ipAddress);
    }

    /**
     * Detaches a role from an instance profile, detaches policies from the role,
     * and deletes all the resources.
     */
    public void deleteInstanceProfile(String roleName, String profileName) {
        try {
            software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
getInstanceProfileRequest =
software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
                    .builder()
                    .instanceProfileName(profileName)
                    .build();

            GetInstanceProfileResponse response =
getIAMClient().getInstanceProfile(getInstanceProfileRequest);
            String name = response.instanceProfile().instanceProfileName();
            System.out.println(name);

            RemoveRoleFromInstanceProfileRequest profileRequest =
RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
                    .roleName(roleName)
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(profileRequest);
            DeleteInstanceProfileRequest deleteInstanceProfileRequest =
DeleteInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
                    .build();

            getIAMClient().deleteInstanceProfile(deleteInstanceProfileRequest);
            System.out.println("Deleted instance profile " + profileName);

            DeleteRoleRequest deleteRoleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();
```

```
            // List attached role policies.
            ListAttachedRolePoliciesResponse rolesResponse = getIAMClient()
                    .listAttachedRolePolicies(role -> role.roleName(roleName));
            List<AttachedPolicy> attachedPolicies =
rolesResponse.attachedPolicies();
            for (AttachedPolicy attachedPolicy : attachedPolicies) {
                DetachRolePolicyRequest request =
DetachRolePolicyRequest.builder()
                        .roleName(roleName)
                        .policyArn(attachedPolicy.policyArn())
                        .build();

                getIAMClient().detachRolePolicy(request);
                System.out.println("Detached and deleted policy " +
attachedPolicy.policyName());
            }

            getIAMClient().deleteRole(deleteRoleRequest);
            System.out.println("Instance profile and role deleted.");

        } catch (IamException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public void deleteTemplate(String templateName) {
        getEc2Client().deleteLaunchTemplate(name ->
name.launchTemplateName(templateName));
        System.out.format(templateName + " was deleted.");
    }

    public void deleteAutoScaleGroup(String groupName) {
        DeleteAutoScalingGroupRequest deleteAutoScalingGroupRequest =
DeleteAutoScalingGroupRequest.builder()
                .autoScalingGroupName(groupName)
                .forceDelete(true)
                .build();


getAutoScalingClient().deleteAutoScalingGroup(deleteAutoScalingGroupRequest);
        System.out.println(groupName + " was deleted.");
    }
```

```
   /*
    * Verify the default security group of the specified VPC allows ingress from
    * this
    * computer. This can be done by allowing ingress from this computer's IP
    * address. In some situations, such as connecting from a corporate network,
you
    * must instead specify a prefix list ID. You can also temporarily open the
port
    * to
    * any IP address while running this example. If you do, be sure to remove
    * public
    * access when you're done.
    *
    */
   public GroupInfo verifyInboundPort(String VPC, int port, String ipAddress) {
       boolean portIsOpen = false;
       GroupInfo groupInfo = new GroupInfo();
       try {
           Filter filter = Filter.builder()
                   .name("group-name")
                   .values("default")
                   .build();

           Filter filter1 = Filter.builder()
                   .name("vpc-id")
                   .values(VPC)
                   .build();

           DescribeSecurityGroupsRequest securityGroupsRequest =
DescribeSecurityGroupsRequest.builder()
                   .filters(filter, filter1)
                   .build();

           DescribeSecurityGroupsResponse securityGroupsResponse =
getEc2Client()
                   .describeSecurityGroups(securityGroupsRequest);
           String securityGroup =
securityGroupsResponse.securityGroups().get(0).groupName();
           groupInfo.setGroupName(securityGroup);

           for (SecurityGroup secGroup :
securityGroupsResponse.securityGroups()) {
```

```
                System.out.println("Found security group: " +
secGroup.groupId());

                for (IpPermission ipPermission : secGroup.ipPermissions()) {
                    if (ipPermission.fromPort() == port) {
                        System.out.println("Found inbound rule: " +
ipPermission);
                        for (IpRange ipRange : ipPermission.ipRanges()) {
                            String cidrIp = ipRange.cidrIp();
                            if (cidrIp.startsWith(ipAddress) ||
cidrIp.equals("0.0.0.0/0")) {
                                System.out.println(cidrIp + " is applicable");
                                portIsOpen = true;
                            }
                        }

                        if (!ipPermission.prefixListIds().isEmpty()) {
                            System.out.println("Prefix lList is applicable");
                            portIsOpen = true;
                        }

                        if (!portIsOpen) {
                            System.out
                                    .println("The inbound rule does not appear to
be open to either this computer's IP,"
                                            + " all IP addresses (0.0.0.0/0), or
to a prefix list ID.");
                        } else {
                            break;
                        }
                    }
                }
            }

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }

        groupInfo.setPortOpen(portIsOpen);
        return groupInfo;
    }

    /*
     * Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
```

```
    * Scaling group.
    * The target group specifies how the load balancer forward requests to the
    * instances
    * in the group.
    */
   public void attachLoadBalancerTargetGroup(String asGroupName, String
targetGroupARN) {
       try {
           AttachLoadBalancerTargetGroupsRequest targetGroupsRequest =
AttachLoadBalancerTargetGroupsRequest.builder()
                   .autoScalingGroupName(asGroupName)
                   .targetGroupARNs(targetGroupARN)
                   .build();


getAutoScalingClient().attachLoadBalancerTargetGroups(targetGroupsRequest);
           System.out.println("Attached load balancer to " + asGroupName);

       } catch (AutoScalingException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   // Creates an EC2 Auto Scaling group with the specified size.
   public String[] createGroup(int groupSize, String templateName, String
autoScalingGroupName) {

       // Get availability zones.

software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
zonesRequest =
software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
               .builder()
               .build();

       DescribeAvailabilityZonesResponse zonesResponse =
getEc2Client().describeAvailabilityZones(zonesRequest);
       List<String> availabilityZoneNames =
zonesResponse.availabilityZones().stream()

.map(software.amazon.awssdk.services.ec2.model.AvailabilityZone::zoneName)
               .collect(Collectors.toList());
```

```
        String availabilityZones = String.join(",", availabilityZoneNames);
        LaunchTemplateSpecification specification =
LaunchTemplateSpecification.builder()
                .launchTemplateName(templateName)
                .version("$Default")
                .build();

        String[] zones = availabilityZones.split(",");
        CreateAutoScalingGroupRequest groupRequest =
CreateAutoScalingGroupRequest.builder()
                .launchTemplate(specification)
                .availabilityZones(zones)
                .maxSize(groupSize)
                .minSize(groupSize)
                .autoScalingGroupName(autoScalingGroupName)
                .build();

        try {
            getAutoScalingClient().createAutoScalingGroup(groupRequest);

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Created an EC2 Auto Scaling group named " +
autoScalingGroupName);
        return zones;
    }

    public String getDefaultVPC() {
        // Define the filter.
        Filter defaultFilter = Filter.builder()
                .name("is-default")
                .values("true")
                .build();

        software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest request =
software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest
                .builder()
                .filters(defaultFilter)
                .build();

        DescribeVpcsResponse response = getEc2Client().describeVpcs(request);
        return response.vpcs().get(0).vpcId();
```

```
    }

    // Gets the default subnets in a VPC for a specified list of Availability
Zones.
    public List<Subnet> getSubnets(String vpcId, String[] availabilityZones) {
        List<Subnet> subnets = null;
        Filter vpcFilter = Filter.builder()
                .name("vpc-id")
                .values(vpcId)
                .build();

        Filter azFilter = Filter.builder()
                .name("availability-zone")
                .values(availabilityZones)
                .build();

        Filter defaultForAZ = Filter.builder()
                .name("default-for-az")
                .values("true")
                .build();

        DescribeSubnetsRequest request = DescribeSubnetsRequest.builder()
                .filters(vpcFilter, azFilter, defaultForAZ)
                .build();

        DescribeSubnetsResponse response =
getEc2Client().describeSubnets(request);
        subnets = response.subnets();
        return subnets;
    }

    // Gets data about the instances in the EC2 Auto Scaling group.
    public String getBadInstance(String groupName) {
        DescribeAutoScalingGroupsRequest request =
DescribeAutoScalingGroupsRequest.builder()
                .autoScalingGroupNames(groupName)
                .build();

        DescribeAutoScalingGroupsResponse response =
getAutoScalingClient().describeAutoScalingGroups(request);
        AutoScalingGroup autoScalingGroup = response.autoScalingGroups().get(0);
        List<String> instanceIds = autoScalingGroup.instances().stream()
                .map(instance -> instance.instanceId())
                .collect(Collectors.toList());
```

```
        String[] instanceIdArray = instanceIds.toArray(new String[0]);
        for (String instanceId : instanceIdArray) {
            System.out.println("Instance ID: " + instanceId);
            return instanceId;
        }
        return "";
    }

    // Gets data about the profile associated with an instance.
    public String getInstanceProfile(String instanceId) {
        Filter filter = Filter.builder()
                .name("instance-id")
                .values(instanceId)
                .build();

        DescribeIamInstanceProfileAssociationsRequest associationsRequest =
DescribeIamInstanceProfileAssociationsRequest
                .builder()
                .filters(filter)
                .build();

        DescribeIamInstanceProfileAssociationsResponse response = getEc2Client()
                .describeIamInstanceProfileAssociations(associationsRequest);
        return response.iamInstanceProfileAssociations().get(0).associationId();
    }

    public void deleteRolesPolicies(String policyName, String roleName, String
InstanceProfile) {
        ListPoliciesRequest listPoliciesRequest =
ListPoliciesRequest.builder().build();
        ListPoliciesResponse listPoliciesResponse =
getIAMClient().listPolicies(listPoliciesRequest);
        for (Policy policy : listPoliciesResponse.policies()) {
            if (policy.policyName().equals(policyName)) {
                // List the entities (users, groups, roles) that are attached to
the policy.

software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
listEntitiesRequest =
software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
                        .builder()
                        .policyArn(policy.arn())
                        .build();
```

```
                ListEntitiesForPolicyResponse listEntitiesResponse = iamClient
                        .listEntitiesForPolicy(listEntitiesRequest);
                if (!listEntitiesResponse.policyGroups().isEmpty() || !
listEntitiesResponse.policyUsers().isEmpty()
                        || !listEntitiesResponse.policyRoles().isEmpty()) {
                    // Detach the policy from any entities it is attached to.
                    DetachRolePolicyRequest detachPolicyRequest =
 DetachRolePolicyRequest.builder()
                            .policyArn(policy.arn())
                            .roleName(roleName) // Specify the name of the IAM
 role
                            .build();

                    getIAMClient().detachRolePolicy(detachPolicyRequest);
                    System.out.println("Policy detached from entities.");
                }

                // Now, you can delete the policy.
                DeletePolicyRequest deletePolicyRequest =
 DeletePolicyRequest.builder()
                        .policyArn(policy.arn())
                        .build();

                getIAMClient().deletePolicy(deletePolicyRequest);
                System.out.println("Policy deleted successfully.");
                break;
            }
        }

        // List the roles associated with the instance profile
        ListInstanceProfilesForRoleRequest listRolesRequest =
 ListInstanceProfilesForRoleRequest.builder()
                .roleName(roleName)
                .build();

        // Detach the roles from the instance profile
        ListInstanceProfilesForRoleResponse listRolesResponse =
 iamClient.listInstanceProfilesForRole(listRolesRequest);
        for (software.amazon.awssdk.services.iam.model.InstanceProfile profile :
 listRolesResponse.instanceProfiles()) {
            RemoveRoleFromInstanceProfileRequest removeRoleRequest =
 RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(InstanceProfile)
                    .roleName(roleName) // Remove the extra dot here
```

```
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(removeRoleRequest);
            System.out.println("Role " + roleName + " removed from instance
 profile " + InstanceProfile);
        }

        // Delete the instance profile after removing all roles
        DeleteInstanceProfileRequest deleteInstanceProfileRequest =
 DeleteInstanceProfileRequest.builder()
                .instanceProfileName(InstanceProfile)
                .build();

        getIAMClient().deleteInstanceProfile(r ->
 r.instanceProfileName(InstanceProfile));
        System.out.println(InstanceProfile + " Deleted");
        System.out.println("All roles and policies are deleted.");
    }
}
```

Créez une classe qui englobe les actions Elastic Load Balancing.

```
public class LoadBalancer {
    public ElasticLoadBalancingV2Client elasticLoadBalancingV2Client;

    public ElasticLoadBalancingV2Client getLoadBalancerClient() {
        if (elasticLoadBalancingV2Client == null) {
            elasticLoadBalancingV2Client = ElasticLoadBalancingV2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }

        return elasticLoadBalancingV2Client;
    }

    // Checks the health of the instances in the target group.
    public List<TargetHealthDescription> checkTargetHealth(String
 targetGroupName) {
        DescribeTargetGroupsRequest targetGroupsRequest =
 DescribeTargetGroupsRequest.builder()
                .names(targetGroupName)
                .build();
```

```
        DescribeTargetGroupsResponse tgResponse =
getLoadBalancerClient().describeTargetGroups(targetGroupsRequest);

        DescribeTargetHealthRequest healthRequest =
DescribeTargetHealthRequest.builder()

.targetGroupArn(tgResponse.targetGroups().get(0).targetGroupArn())
                .build();

        DescribeTargetHealthResponse healthResponse =
getLoadBalancerClient().describeTargetHealth(healthRequest);
        return healthResponse.targetHealthDescriptions();
    }

    // Gets the HTTP endpoint of the load balancer.
    public String getEndpoint(String lbName) {
        DescribeLoadBalancersResponse res = getLoadBalancerClient()
                .describeLoadBalancers(describe -> describe.names(lbName));
        return res.loadBalancers().get(0).dnsName();
    }

    // Deletes a load balancer.
    public void deleteLoadBalancer(String lbName) {
        try {
            // Use a waiter to delete the Load Balancer.
            DescribeLoadBalancersResponse res = getLoadBalancerClient()
                    .describeLoadBalancers(describe -> describe.names(lbName));
            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()

.loadBalancerArns(res.loadBalancers().get(0).loadBalancerArn())
                    .build();

            getLoadBalancerClient().deleteLoadBalancer(
                    builder ->
builder.loadBalancerArn(res.loadBalancers().get(0).loadBalancerArn()));
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                    .waitUntilLoadBalancersDeleted(request);
            waiterResponse.matched().response().ifPresent(System.out::println);
```

```
        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(lbName + " was deleted.");
    }

    // Deletes the target group.
    public void deleteTargetGroup(String targetGroupName) {
        try {
            DescribeTargetGroupsResponse res = getLoadBalancerClient()
                    .describeTargetGroups(describe ->
describe.names(targetGroupName));
            getLoadBalancerClient()
                    .deleteTargetGroup(builder ->
builder.targetGroupArn(res.targetGroups().get(0).targetGroupArn()));
        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(targetGroupName + " was deleted.");
    }

    // Verify this computer can successfully send a GET request to the load
balancer
    // endpoint.
    public boolean verifyLoadBalancerEndpoint(String elbDnsName) throws
IOException, InterruptedException {
        boolean success = false;
        int retries = 3;
        CloseableHttpClient httpClient = HttpClients.createDefault();

        // Create an HTTP GET request to the ELB.
        HttpGet httpGet = new HttpGet("http://" + elbDnsName);
        try {
            while ((!success) && (retries > 0)) {
                // Execute the request and get the response.
                HttpResponse response = httpClient.execute(httpGet);
                int statusCode = response.getStatusLine().getStatusCode();
                System.out.println("HTTP Status Code: " + statusCode);
                if (statusCode == 200) {
                    success = true;
                } else {
                    retries--;
                    System.out.println("Got connection error from load balancer
endpoint, retrying...");
```

```
                    TimeUnit.SECONDS.sleep(15);
                }
            }

        } catch (org.apache.http.conn.HttpHostConnectException e) {
            System.out.println(e.getMessage());
        }

        System.out.println("Status.." + success);
        return success;
    }

    /*
     * Creates an Elastic Load Balancing target group. The target group specifies
     * how
     * the load balancer forward requests to instances in the group and how
instance
     * health is checked.
     */
    public String createTargetGroup(String protocol, int port, String vpcId,
String targetGroupName) {
        CreateTargetGroupRequest targetGroupRequest =
CreateTargetGroupRequest.builder()
                .healthCheckPath("/healthcheck")
                .healthCheckTimeoutSeconds(5)
                .port(port)
                .vpcId(vpcId)
                .name(targetGroupName)
                .protocol(protocol)
                .build();

        CreateTargetGroupResponse targetGroupResponse =
getLoadBalancerClient().createTargetGroup(targetGroupRequest);
        String targetGroupArn =
targetGroupResponse.targetGroups().get(0).targetGroupArn();
        String targetGroup =
targetGroupResponse.targetGroups().get(0).targetGroupName();
        System.out.println("The " + targetGroup + " was created with ARN" +
targetGroupArn);
        return targetGroupArn;
    }

    /*
     * Creates an Elastic Load Balancing load balancer that uses the specified
```

```
    * subnets
    * and forwards requests to the specified target group.
    */
   public String createLoadBalancer(List<Subnet> subnetIds, String
targetGroupARN, String lbName, int port,
           String protocol) {
       try {
           List<String> subnetIdStrings = subnetIds.stream()
                   .map(Subnet::subnetId)
                   .collect(Collectors.toList());

           CreateLoadBalancerRequest balancerRequest =
CreateLoadBalancerRequest.builder()
                   .subnets(subnetIdStrings)
                   .name(lbName)
                   .scheme("internet-facing")
                   .build();

           // Create and wait for the load balancer to become available.
           CreateLoadBalancerResponse lsResponse =
getLoadBalancerClient().createLoadBalancer(balancerRequest);
           String lbARN = lsResponse.loadBalancers().get(0).loadBalancerArn();

           ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
           DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()
                   .loadBalancerArns(lbARN)
                   .build();

           System.out.println("Waiting for Load Balancer " + lbName + " to
become available.");
           WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                   .waitUntilLoadBalancerAvailable(request);
           waiterResponse.matched().response().ifPresent(System.out::println);
           System.out.println("Load Balancer " + lbName + " is available.");

           // Get the DNS name (endpoint) of the load balancer.
           String lbDNSName = lsResponse.loadBalancers().get(0).dnsName();
           System.out.println("*** Load Balancer DNS Name: " + lbDNSName);

           // Create a listener for the load balance.
           Action action = Action.builder()
```

```
                    .targetGroupArn(targetGroupARN)
                    .type("forward")
                    .build();

            CreateListenerRequest listenerRequest =
 CreateListenerRequest.builder()

 .loadBalancerArn(lsResponse.loadBalancers().get(0).loadBalancerArn())
                    .defaultActions(action)
                    .port(port)
                    .protocol(protocol)
                    .build();

            getLoadBalancerClient().createListener(listenerRequest);
            System.out.println("Created listener to forward traffic from load
 balancer " + lbName + " to target group "
                    + targetGroupARN);

            // Return the load balancer DNS name.
            return lbDNSName;

        } catch (ElasticLoadBalancingV2Exception e) {
            e.printStackTrace();
        }
        return "";
    }
}
```

Créez une classe qui utilise DynamoDB pour simuler un service de recommandation.

```
public class Database {

    private static DynamoDbClient dynamoDbClient;

    public static DynamoDbClient getDynamoDbClient() {
        if (dynamoDbClient == null) {
            dynamoDbClient = DynamoDbClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return dynamoDbClient;
    }
```

```
   // Checks to see if the Amazon DynamoDB table exists.
   private boolean doesTableExist(String tableName) {
       try {
           // Describe the table and catch any exceptions.
           DescribeTableRequest describeTableRequest =
DescribeTableRequest.builder()
                   .tableName(tableName)
                   .build();

           getDynamoDbClient().describeTable(describeTableRequest);
           System.out.println("Table '" + tableName + "' exists.");
           return true;

       } catch (ResourceNotFoundException e) {
           System.out.println("Table '" + tableName + "' does not exist.");
       } catch (DynamoDbException e) {
           System.err.println("Error checking table existence: " +
e.getMessage());
       }
       return false;
   }

   /*
    * Creates a DynamoDB table to use a recommendation service. The table has a
    * hash key named 'MediaType' that defines the type of media recommended,
such
    * as
    * Book or Movie, and a range key named 'ItemId' that, combined with the
    * MediaType,
    * forms a unique identifier for the recommended item.
    */
   public void createTable(String tableName, String fileName) throws IOException
{
       // First check to see if the table exists.
       boolean doesExist = doesTableExist(tableName);
       if (!doesExist) {
           DynamoDbWaiter dbWaiter = getDynamoDbClient().waiter();
           CreateTableRequest createTableRequest = CreateTableRequest.builder()
                   .tableName(tableName)
                   .attributeDefinitions(
                           AttributeDefinition.builder()
                                   .attributeName("MediaType")
                                   .attributeType(ScalarAttributeType.S)
```

```
                                .build(),
                        AttributeDefinition.builder()
                                .attributeName("ItemId")
                                .attributeType(ScalarAttributeType.N)
                                .build())
                .keySchema(
                        KeySchemaElement.builder()
                                .attributeName("MediaType")
                                .keyType(KeyType.HASH)
                                .build(),
                        KeySchemaElement.builder()
                                .attributeName("ItemId")
                                .keyType(KeyType.RANGE)
                                .build())
                .provisionedThroughput(
                        ProvisionedThroughput.builder()
                                .readCapacityUnits(5L)
                                .writeCapacityUnits(5L)
                                .build())
                .build();

        getDynamoDbClient().createTable(createTableRequest);
        System.out.println("Creating table " + tableName + "...");

        // Wait until the Amazon DynamoDB table is created.
        DescribeTableRequest tableRequest = DescribeTableRequest.builder()
                .tableName(tableName)
                .build();

        WaiterResponse<DescribeTableResponse> waiterResponse =
dbWaiter.waitUntilTableExists(tableRequest);
        waiterResponse.matched().response().ifPresent(System.out::println);
        System.out.println("Table " + tableName + " created.");

        // Add records to the table.
        populateTable(fileName, tableName);
    }
}

public void deleteTable(String tableName) {
    getDynamoDbClient().deleteTable(table -> table.tableName(tableName));
    System.out.println("Table " + tableName + " deleted.");
}
```

```
    // Populates the table with data located in a JSON file using the DynamoDB
    // enhanced client.
    public void populateTable(String fileName, String tableName) throws
 IOException {
        DynamoDbEnhancedClient enhancedClient = DynamoDbEnhancedClient.builder()
                .dynamoDbClient(getDynamoDbClient())
                .build();
        ObjectMapper objectMapper = new ObjectMapper();
        File jsonFile = new File(fileName);
        JsonNode rootNode = objectMapper.readTree(jsonFile);

        DynamoDbTable<Recommendation> mappedTable =
 enhancedClient.table(tableName,
                TableSchema.fromBean(Recommendation.class));
        for (JsonNode currentNode : rootNode) {
            String mediaType = currentNode.path("MediaType").path("S").asText();
            int itemId = currentNode.path("ItemId").path("N").asInt();
            String title = currentNode.path("Title").path("S").asText();
            String creator = currentNode.path("Creator").path("S").asText();

            // Create a Recommendation object and set its properties.
            Recommendation rec = new Recommendation();
            rec.setMediaType(mediaType);
            rec.setItemId(itemId);
            rec.setTitle(title);
            rec.setCreator(creator);

            // Put the item into the DynamoDB table.
            mappedTable.putItem(rec); // Add the Recommendation to the list.
        }
        System.out.println("Added all records to the " + tableName);
    }
}
```

Créez une classe qui englobe les actions de Systems Manager.

```
public class ParameterHelper {

    String tableName = "doc-example-resilient-architecture-table";
    String dyntable = "doc-example-recommendation-service";
    String failureResponse = "doc-example-resilient-architecture-failure-
response";
```

```
    String healthCheck = "doc-example-resilient-architecture-health-check";

    public void reset() {
        put(dyntable, tableName);
        put(failureResponse, "none");
        put(healthCheck, "shallow");
    }

    public void put(String name, String value) {
        SsmClient ssmClient = SsmClient.builder()
                .region(Region.US_EAST_1)
                .build();

        PutParameterRequest parameterRequest = PutParameterRequest.builder()
                .name(name)
                .value(value)
                .overwrite(true)
                .type("String")
                .build();

        ssmClient.putParameter(parameterRequest);
        System.out.printf("Setting demo parameter %s to '%s'.", name, value);
    }
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la référence de l'API AWS SDK for Java 2.x .
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer

- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Exécutez un scénario interactif à une invite de commande.

```
#!/usr/bin/env node
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0

import {
  Scenario,
  parseScenarioArgs,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
```

```
/**
 * The workflow steps are split into three stages:
 *   - deploy
 *   - demo
 *   - destroy
 *
 * Each of these stages has a corresponding file prefixed with steps-*.
 */
import { deploySteps } from "./steps-deploy.js";
import { demoSteps } from "./steps-demo.js";
import { destroySteps } from "./steps-destroy.js";

/**
 * The context is passed to every scenario. Scenario steps
 * will modify the context.
 */
const context = {};

/**
 * Three Scenarios are created for the workflow. A Scenario is an orchestration
 class
 * that simplifies running a series of steps.
 */
export const scenarios = {
  // Deploys all resources necessary for the workflow.
  deploy: new Scenario("Resilient Workflow - Deploy", deploySteps, context),
  // Demonstrates how a fragile web service can be made more resilient.
  demo: new Scenario("Resilient Workflow - Demo", demoSteps, context),
  // Destroys the resources created for the workflow.
  destroy: new Scenario("Resilient Workflow - Destroy", destroySteps, context),
};

// Call function if run directly
import { fileURLToPath } from "node:url";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios, {
    name: "Resilient Workflow",
    synopsis:
      "node index.js --scenario <deploy | demo | destroy> [-h|--help] [-y|--yes]
 [-v|--verbose]",
    description: "Deploy and interact with scalable EC2 instances.",
  });
}
```

Créez des étapes pour déployer toutes les ressources.

```
import { join } from "node:path";
import { readFileSync, writeFileSync } from "node:fs";
import axios from "axios";

import {
  BatchWriteItemCommand,
  CreateTableCommand,
  DynamoDBClient,
  waitUntilTableExists,
} from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  CreateKeyPairCommand,
  CreateLaunchTemplateCommand,
  DescribeAvailabilityZonesCommand,
  DescribeVpcsCommand,
  DescribeSubnetsCommand,
  DescribeSecurityGroupsCommand,
  AuthorizeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  AttachRolePolicyCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import { SSMClient, GetParameterCommand } from "@aws-sdk/client-ssm";
import {
  CreateAutoScalingGroupCommand,
  AutoScalingClient,
  AttachLoadBalancerTargetGroupsCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  CreateListenerCommand,
  CreateLoadBalancerCommand,
  CreateTargetGroupCommand,
```

```
  ElasticLoadBalancingV2Client,
  waitUntilLoadBalancerAvailable,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { saveState } from "@aws-doc-sdk-examples/lib/scenario/steps-common.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH, ROOT } from "./constants.js";
import { initParamsSteps } from "./steps-reset-params.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const deploySteps = [
  new ScenarioOutput("introduction", MESSAGES.introduction, { header: true }),
  new ScenarioInput("confirmDeployment", MESSAGES.confirmDeployment, {
    type: "confirm",
  }),
  new ScenarioAction(
    "handleConfirmDeployment",
    (c) => c.confirmDeployment === false && process.exit(),
  ),
  new ScenarioOutput(
    "creatingTable",
    MESSAGES.creatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("createTable", async () => {
    const client = new DynamoDBClient({});
    await client.send(
      new CreateTableCommand({
        TableName: NAMES.tableName,
        ProvisionedThroughput: {
          ReadCapacityUnits: 5,
          WriteCapacityUnits: 5,
        },
        AttributeDefinitions: [
          {
            AttributeName: "MediaType",
            AttributeType: "S",
```

```
          },
          {
            AttributeName: "ItemId",
            AttributeType: "N",
          },
        ],
        KeySchema: [
          {
            AttributeName: "MediaType",
            KeyType: "HASH",
          },
          {
            AttributeName: "ItemId",
            KeyType: "RANGE",
          },
        ],
      }),
    );
    await waitUntilTableExists({ client }, { TableName: NAMES.tableName });
  }),
  new ScenarioOutput(
    "createdTable",
    MESSAGES.createdTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "populatingTable",
    MESSAGES.populatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("populateTable", () => {
    const client = new DynamoDBClient({});
    /**
     * @type {{ default: import("@aws-sdk/client-dynamodb").PutRequest['Item']
[] }}
     */
    const recommendations = JSON.parse(
      readFileSync(join(RESOURCES_PATH, "recommendations.json")),
    );

    return client.send(
      new BatchWriteItemCommand({
        RequestItems: {
          [NAMES.tableName]: recommendations.map((item) => ({
            PutRequest: { Item: item },
          })),
```

```
      },
    }),
  );
}),
new ScenarioOutput(
  "populatedTable",
  MESSAGES.populatedTable.replace("${TABLE_NAME}", NAMES.tableName),
),
new ScenarioOutput(
  "creatingKeyPair",
  MESSAGES.creatingKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
),
new ScenarioAction("createKeyPair", async () => {
  const client = new EC2Client({});
  const { KeyMaterial } = await client.send(
    new CreateKeyPairCommand({
      KeyName: NAMES.keyPairName,
    }),
  );

  writeFileSync(`${NAMES.keyPairName}.pem`, KeyMaterial, { mode: 0o600 });
}),
new ScenarioOutput(
  "createdKeyPair",
  MESSAGES.createdKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
),
new ScenarioOutput(
  "creatingInstancePolicy",
  MESSAGES.creatingInstancePolicy.replace(
    "${INSTANCE_POLICY_NAME}",
    NAMES.instancePolicyName,
  ),
),
new ScenarioAction("createInstancePolicy", async (state) => {
  const client = new IAMClient({});
  const {
    Policy: { Arn },
  } = await client.send(
    new CreatePolicyCommand({
      PolicyName: NAMES.instancePolicyName,
      PolicyDocument: readFileSync(
        join(RESOURCES_PATH, "instance_policy.json"),
      ),
    }),
```

```
    );
    state.instancePolicyArn = Arn;
  }),
  new ScenarioOutput("createdInstancePolicy", (state) =>
    MESSAGES.createdInstancePolicy
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_POLICY_ARN}", state.instancePolicyArn),
  ),
  new ScenarioOutput(
    "creatingInstanceRole",
    MESSAGES.creatingInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioAction("createInstanceRole", () => {
    const client = new IAMClient({});
    return client.send(
      new CreateRoleCommand({
        RoleName: NAMES.instanceRoleName,
        AssumeRolePolicyDocument: readFileSync(
          join(ROOT, "assume-role-policy.json"),
        ),
      }),
    );
  }),
  new ScenarioOutput(
    "createdInstanceRole",
    MESSAGES.createdInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioOutput(
    "attachingPolicyToRole",
    MESSAGES.attachingPolicyToRole
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName)
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName),
  ),
  new ScenarioAction("attachPolicyToRole", async (state) => {
    const client = new IAMClient({});
    await client.send(
      new AttachRolePolicyCommand({
        RoleName: NAMES.instanceRoleName,
```

```
      PolicyArn: state.instancePolicyArn,
    }),
  );
}),
new ScenarioOutput(
  "attachedPolicyToRole",
  MESSAGES.attachedPolicyToRole
    .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
    .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
),
new ScenarioOutput(
  "creatingInstanceProfile",
  MESSAGES.creatingInstanceProfile.replace(
    "${INSTANCE_PROFILE_NAME}",
    NAMES.instanceProfileName,
  ),
),
new ScenarioAction("createInstanceProfile", async (state) => {
  const client = new IAMClient({});
  const {
    InstanceProfile: { Arn },
  } = await client.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.instanceProfileName,
    }),
  );
  state.instanceProfileArn = Arn;

  await waitUntilInstanceProfileExists(
    { client },
    { InstanceProfileName: NAMES.instanceProfileName },
  );
}),
new ScenarioOutput("createdInstanceProfile", (state) =>
  MESSAGES.createdInstanceProfile
    .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
    .replace("${INSTANCE_PROFILE_ARN}", state.instanceProfileArn),
),
new ScenarioOutput(
  "addingRoleToInstanceProfile",
  MESSAGES.addingRoleToInstanceProfile
    .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
    .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
),
```

```
  new ScenarioAction("addRoleToInstanceProfile", () => {
    const client = new IAMClient({});
    return client.send(
      new AddRoleToInstanceProfileCommand({
        RoleName: NAMES.instanceRoleName,
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
  }),
  new ScenarioOutput(
    "addedRoleToInstanceProfile",
    MESSAGES.addedRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  ...initParamsSteps,
  new ScenarioOutput("creatingLaunchTemplate", MESSAGES.creatingLaunchTemplate),
  new ScenarioAction("createLaunchTemplate", async () => {
    const ssmClient = new SSMClient({});
    const { Parameter } = await ssmClient.send(
      new GetParameterCommand({
        Name: "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
      }),
    );
    const ec2Client = new EC2Client({});
    await ec2Client.send(
      new CreateLaunchTemplateCommand({
        LaunchTemplateName: NAMES.launchTemplateName,
        LaunchTemplateData: {
          InstanceType: "t3.micro",
          ImageId: Parameter.Value,
          IamInstanceProfile: { Name: NAMES.instanceProfileName },
          UserData: readFileSync(
            join(RESOURCES_PATH, "server_startup_script.sh"),
          ).toString("base64"),
          KeyName: NAMES.keyPairName,
        },
      }),
    );
  }),
  new ScenarioOutput(
    "createdLaunchTemplate",
    MESSAGES.createdLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
```

```
      NAMES.launchTemplateName,
    ),
  ),
  new ScenarioOutput(
    "creatingAutoScalingGroup",
    MESSAGES.creatingAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    ),
  ),
  new ScenarioAction("createAutoScalingGroup", async (state) => {
    const ec2Client = new EC2Client({});
    const { AvailabilityZones } = await ec2Client.send(
      new DescribeAvailabilityZonesCommand({}),
    );
    state.availabilityZoneNames = AvailabilityZones.map((az) => az.ZoneName);
    const autoScalingClient = new AutoScalingClient({});
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new CreateAutoScalingGroupCommand({
          AvailabilityZones: state.availabilityZoneNames,
          AutoScalingGroupName: NAMES.autoScalingGroupName,
          LaunchTemplate: {
            LaunchTemplateName: NAMES.launchTemplateName,
            Version: "$Default",
          },
          MinSize: 3,
          MaxSize: 3,
        }),
      ),
    );
  }),
  new ScenarioOutput(
    "createdAutoScalingGroup",
    /**
     * @param {{ availabilityZoneNames: string[] }} state
     */
    (state) =>
      MESSAGES.createdAutoScalingGroup
        .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName)
        .replace(
          "${AVAILABILITY_ZONE_NAMES}",
          state.availabilityZoneNames.join(", "),
        ),
```

```
  ),
  new ScenarioInput("confirmContinue", MESSAGES.confirmContinue, {
    type: "confirm",
  }),
  new ScenarioOutput("loadBalancer", MESSAGES.loadBalancer),
  new ScenarioOutput("gettingVpc", MESSAGES.gettingVpc),
  new ScenarioAction("getVpc", async (state) => {
    const client = new EC2Client({});
    const { Vpcs } = await client.send(
      new DescribeVpcsCommand({
        Filters: [{ Name: "is-default", Values: ["true"] }],
      }),
    );
    state.defaultVpc = Vpcs[0].VpcId;
  }),
  new ScenarioOutput("gotVpc", (state) =>
    MESSAGES.gotVpc.replace("${VPC_ID}", state.defaultVpc),
  ),
  new ScenarioOutput("gettingSubnets", MESSAGES.gettingSubnets),
  new ScenarioAction("getSubnets", async (state) => {
    const client = new EC2Client({});
    const { Subnets } = await client.send(
      new DescribeSubnetsCommand({
        Filters: [
          { Name: "vpc-id", Values: [state.defaultVpc] },
          { Name: "availability-zone", Values: state.availabilityZoneNames },
          { Name: "default-for-az", Values: ["true"] },
        ],
      }),
    );
    state.subnets = Subnets.map((subnet) => subnet.SubnetId);
  }),
  new ScenarioOutput(
    "gotSubnets",
    /**
     * @param {{ subnets: string[] }} state
     */
    (state) =>
      MESSAGES.gotSubnets.replace("${SUBNETS}", state.subnets.join(", ")),
  ),
  new ScenarioOutput(
    "creatingLoadBalancerTargetGroup",
    MESSAGES.creatingLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
```

```
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioAction("createLoadBalancerTargetGroup", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { TargetGroups } = await client.send(
      new CreateTargetGroupCommand({
        Name: NAMES.loadBalancerTargetGroupName,
        Protocol: "HTTP",
        Port: 80,
        HealthCheckPath: "/healthcheck",
        HealthCheckIntervalSeconds: 10,
        HealthCheckTimeoutSeconds: 5,
        HealthyThresholdCount: 2,
        UnhealthyThresholdCount: 2,
        VpcId: state.defaultVpc,
      }),
    );
    const targetGroup = TargetGroups[0];
    state.targetGroupArn = targetGroup.TargetGroupArn;
    state.targetGroupProtocol = targetGroup.Protocol;
    state.targetGroupPort = targetGroup.Port;
  }),
  new ScenarioOutput(
    "createdLoadBalancerTargetGroup",
    MESSAGES.createdLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioOutput(
    "creatingLoadBalancer",
    MESSAGES.creatingLoadBalancer.replace("${LB_NAME}", NAMES.loadBalancerName),
  ),
  new ScenarioAction("createLoadBalancer", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { LoadBalancers } = await client.send(
      new CreateLoadBalancerCommand({
        Name: NAMES.loadBalancerName,
        Subnets: state.subnets,
      }),
    );
    state.loadBalancerDns = LoadBalancers[0].DNSName;
    state.loadBalancerArn = LoadBalancers[0].LoadBalancerArn;
```

```
    await waitUntilLoadBalancerAvailable(
      { client },
      { Names: [NAMES.loadBalancerName] },
    );
  }),
  new ScenarioOutput("createdLoadBalancer", (state) =>
    MESSAGES.createdLoadBalancer
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioOutput(
    "creatingListener",
    MESSAGES.creatingLoadBalancerListener
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName),
  ),
  new ScenarioAction("createListener", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { Listeners } = await client.send(
      new CreateListenerCommand({
        LoadBalancerArn: state.loadBalancerArn,
        Protocol: state.targetGroupProtocol,
        Port: state.targetGroupPort,
        DefaultActions: [
          { Type: "forward", TargetGroupArn: state.targetGroupArn },
        ],
      }),
    );
    const listener = Listeners[0];
    state.loadBalancerListenerArn = listener.ListenerArn;
  }),
  new ScenarioOutput("createdListener", (state) =>
    MESSAGES.createdLoadBalancerListener.replace(
      "${LB_LISTENER_ARN}",
      state.loadBalancerListenerArn,
    ),
  ),
  new ScenarioOutput(
    "attachingLoadBalancerTargetGroup",
    MESSAGES.attachingLoadBalancerTargetGroup
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName)
      .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName),
  ),
  new ScenarioAction("attachLoadBalancerTargetGroup", async (state) => {
```

```
    const client = new AutoScalingClient({});
    await client.send(
      new AttachLoadBalancerTargetGroupsCommand({
        AutoScalingGroupName: NAMES.autoScalingGroupName,
        TargetGroupARNs: [state.targetGroupArn],
      }),
    );
  }),
  new ScenarioOutput(
    "attachedLoadBalancerTargetGroup",
    MESSAGES.attachedLoadBalancerTargetGroup,
  ),
  new ScenarioOutput("verifyingInboundPort", MESSAGES.verifyingInboundPort),
  new ScenarioAction(
    "verifyInboundPort",
    /**
     *
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup}} state
     */
    async (state) => {
      const client = new EC2Client({});
      const { SecurityGroups } = await client.send(
        new DescribeSecurityGroupsCommand({
          Filters: [{ Name: "group-name", Values: ["default"] }],
        }),
      );
      if (!SecurityGroups) {
        state.verifyInboundPortError = new Error(MESSAGES.noSecurityGroups);
      }
      state.defaultSecurityGroup = SecurityGroups[0];

      /**
       * @type {string}
       */
      const ipResponse = (await axios.get("http://checkip.amazonaws.com")).data;
      state.myIp = ipResponse.trim();
      const myIpRules = state.defaultSecurityGroup.IpPermissions.filter(
        ({ IpRanges }) =>
          IpRanges.some(
            ({ CidrIp }) =>
              CidrIp.startsWith(state.myIp) || CidrIp === "0.0.0.0/0",
          ),
      )
```

```
        .filter(({ IpProtocol }) => IpProtocol === "tcp")
        .filter(({ FromPort }) => FromPort === 80);

      state.myIpRules = myIpRules;
    },
  ),
  new ScenarioOutput(
    "verifiedInboundPort",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return MESSAGES.foundIpRules.replace(
          "${IP_RULES}",
          JSON.stringify(state.myIpRules, null, 2),
        );
      }
      return MESSAGES.noIpRules;
    },
  ),
  new ScenarioInput(
    "shouldAddInboundRule",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return false;
      }
      return MESSAGES.noIpRules;
    },
    { type: "confirm" },
  ),
  new ScenarioAction(
    "addInboundRule",
    /**
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup }} state
     */
    async (state) => {
      if (!state.shouldAddInboundRule) {
        return;
      }
```

```
      const client = new EC2Client({});
      await client.send(
        new AuthorizeSecurityGroupIngressCommand({
          GroupId: state.defaultSecurityGroup.GroupId,
          CidrIp: `${state.myIp}/32`,
          FromPort: 80,
          ToPort: 80,
          IpProtocol: "tcp",
        }),
      );
    },
  ),
  new ScenarioOutput("addedInboundRule", (state) => {
    if (state.shouldAddInboundRule) {
      return MESSAGES.addedInboundRule.replace("${IP_ADDRESS}", state.myIp);
    }
    return false;
  }),
  new ScenarioOutput("verifyingEndpoint", (state) =>
    MESSAGES.verifyingEndpoint.replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioAction("verifyEndpoint", async (state) => {
    try {
      const response = await retry({ intervalInMs: 2000, maxRetries: 30 }, () =>
        axios.get(`http://${state.loadBalancerDns}`),
      );
      state.endpointResponse = JSON.stringify(response.data, null, 2);
    } catch (e) {
      state.verifyEndpointError = e;
    }
  }),
  new ScenarioOutput("verifiedEndpoint", (state) => {
    if (state.verifyEndpointError) {
      console.error(state.verifyEndpointError);
    } else {
      return MESSAGES.verifiedEndpoint.replace(
        "${ENDPOINT_RESPONSE}",
        state.endpointResponse,
      );
    }
  }),
  saveState,
];
```

Créez des étapes pour exécuter la démo.

```
import { readFileSync } from "node:fs";
import { join } from "node:path";

import axios from "axios";

import {
  DescribeTargetGroupsCommand,
  DescribeTargetHealthCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";
import {
  DescribeInstanceInformationCommand,
  PutParameterCommand,
  SSMClient,
  SendCommandCommand,
} from "@aws-sdk/client-ssm";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DescribeAutoScalingGroupsCommand,
  TerminateInstanceInAutoScalingGroupCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  DescribeIamInstanceProfileAssociationsCommand,
  EC2Client,
  RebootInstancesCommand,
  ReplaceIamInstanceProfileAssociationCommand,
} from "@aws-sdk/client-ec2";

import {
  ScenarioAction,
```

```
  ScenarioInput,
  ScenarioOutput,
} from "@aws-doc-sdk-examples/lib/scenario/scenario.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

const getRecommendation = new ScenarioAction(
  "getRecommendation",
  async (state) => {
    const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
    if (loadBalancer) {
      state.loadBalancerDnsName = loadBalancer.DNSName;
      try {
        state.recommendation = (
          await axios.get(`http://${state.loadBalancerDnsName}`)
        ).data;
      } catch (e) {
        state.recommendation = e instanceof Error ? e.message : e;
      }
    } else {
      throw new Error(MESSAGES.demoFindLoadBalancerError);
    }
  },
);

const getRecommendationResult = new ScenarioOutput(
  "getRecommendationResult",
  (state) =>
    `Recommendation:\n${JSON.stringify(state.recommendation, null, 2)}`,
  { preformatted: true },
);

const getHealthCheck = new ScenarioAction("getHealthCheck", async (state) => {
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
    new DescribeTargetGroupsCommand({
      Names: [NAMES.loadBalancerTargetGroupName],
    }),
  );

  const { TargetHealthDescriptions } = await client.send(
    new DescribeTargetHealthCommand({
```

```
      TargetGroupArn: TargetGroups[0].TargetGroupArn,
    }),
  );
  state.targetHealthDescriptions = TargetHealthDescriptions;
});

const getHealthCheckResult = new ScenarioOutput(
  "getHealthCheckResult",
  /**
   * @param {{ targetHealthDescriptions: import('@aws-sdk/client-elastic-load-
balancing-v2').TargetHealthDescription[]}} state
   */
  (state) => {
    const status = state.targetHealthDescriptions
      .map((th) => `${th.Target.Id}: ${th.TargetHealth.State}`)
      .join("\n");
    return `Health check:\n${status}`;
  },
  { preformatted: true },
);

const loadBalancerLoop = new ScenarioAction(
  "loadBalancerLoop",
  getRecommendation.action,
  {
    whileConfig: {
      whileFn: ({ loadBalancerCheck }) => loadBalancerCheck,
      input: new ScenarioInput(
        "loadBalancerCheck",
        MESSAGES.demoLoadBalancerCheck,
        {
          type: "confirm",
        },
      ),
      output: getRecommendationResult,
    },
  },
);

const healthCheckLoop = new ScenarioAction(
  "healthCheckLoop",
  getHealthCheck.action,
  {
    whileConfig: {
```

```
      whileFn: ({ healthCheck }) => healthCheck,
      input: new ScenarioInput("healthCheck", MESSAGES.demoHealthCheck, {
        type: "confirm",
      }),
      output: getHealthCheckResult,
    },
  },
);

const statusSteps = [
  getRecommendation,
  getRecommendationResult,
  getHealthCheck,
  getHealthCheckResult,
];

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const demoSteps = [
  new ScenarioOutput("header", MESSAGES.demoHeader, { header: true }),
  new ScenarioOutput("sanityCheck", MESSAGES.demoSanityCheck),
  ...statusSteps,
  new ScenarioInput(
    "brokenDependencyConfirmation",
    MESSAGES.demoBrokenDependencyConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("brokenDependency", async (state) => {
    if (!state.brokenDependencyConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      state.badTableName = `fake-table-${Date.now()}`;
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmTableNameKey,
          Value: state.badTableName,
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
```

```
  new ScenarioOutput("testBrokenDependency", (state) =>
    MESSAGES.demoTestBrokenDependency.replace(
      "${TABLE_NAME}",
      state.badTableName,
    ),
  ),
  ...statusSteps,
  new ScenarioInput(
    "staticResponseConfirmation",
    MESSAGES.demoStaticResponseConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("staticResponse", async (state) => {
    if (!state.staticResponseConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmFailureResponseKey,
          Value: "static",
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testStaticResponse", MESSAGES.demoTestStaticResponse),
  ...statusSteps,
  new ScenarioInput(
    "badCredentialsConfirmation",
    MESSAGES.demoBadCredentialsConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("badCredentialsExit", (state) => {
    if (!state.badCredentialsConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("fixDynamoDBName", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
```

```
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioAction(
    "badCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-auto-
scaling').Instance }} state
     */
    async (state) => {
      await createSsmOnlyInstanceProfile();
      const autoScalingClient = new AutoScalingClient({});
      const { AutoScalingGroups } = await autoScalingClient.send(
        new DescribeAutoScalingGroupsCommand({
          AutoScalingGroupNames: [NAMES.autoScalingGroupName],
        }),
      );
      state.targetInstance = AutoScalingGroups[0].Instances[0];
      const ec2Client = new EC2Client({});
      const { IamInstanceProfileAssociations } = await ec2Client.send(
        new DescribeIamInstanceProfileAssociationsCommand({
          Filters: [
            { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
          ],
        }),
      );
      state.instanceProfileAssociationId =
        IamInstanceProfileAssociations[0].AssociationId;
      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        ec2Client.send(
          new ReplaceIamInstanceProfileAssociationCommand({
            AssociationId: state.instanceProfileAssociationId,
            IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
          }),
        ),
      );

      await ec2Client.send(
        new RebootInstancesCommand({
          InstanceIds: [state.targetInstance.InstanceId],
        }),
```

```
      );

      const ssmClient = new SSMClient({});
      await retry({ intervalInMs: 20000, maxRetries: 15 }, async () => {
        const { InstanceInformationList } = await ssmClient.send(
          new DescribeInstanceInformationCommand({}),
        );

        const instance = InstanceInformationList.find(
          (info) => info.InstanceId === state.targetInstance.InstanceId,
        );

        if (!instance) {
          throw new Error("Instance not found.");
        }
      });

      await ssmClient.send(
        new SendCommandCommand({
          InstanceIds: [state.targetInstance.InstanceId],
          DocumentName: "AWS-RunShellScript",
          Parameters: { commands: ["cd / && sudo python3 server.py 80"] },
        }),
      );
    },
  ),
  new ScenarioOutput(
    "testBadCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation}} state
     */
    (state) =>
      MESSAGES.demoTestBadCredentials.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
  ),
  loadBalancerLoop,
  new ScenarioInput(
    "deepHealthCheckConfirmation",
    MESSAGES.demoDeepHealthCheckConfirmation,
    { type: "confirm" },
  ),
```

```
  new ScenarioAction("deepHealthCheckExit", (state) => {
    if (!state.deepHealthCheckConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("deepHealthCheck", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmHealthCheckKey,
        Value: "deep",
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testDeepHealthCheck", MESSAGES.demoTestDeepHealthCheck),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "killInstanceConfirmation",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    (state) =>
      MESSAGES.demoKillInstanceConfirmation.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
    { type: "confirm" },
  ),
  new ScenarioAction("killInstanceExit", (state) => {
    if (!state.killInstanceConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction(
    "killInstance",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    async (state) => {
```

```
      const client = new AutoScalingClient({});
      await client.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: state.targetInstance.InstanceId,
          ShouldDecrementDesiredCapacity: false,
        }),
      );
    },
  ),
  new ScenarioOutput("testKillInstance", MESSAGES.demoTestKillInstance),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput("failOpenConfirmation", MESSAGES.demoFailOpenConfirmation, {
    type: "confirm",
  }),
  new ScenarioAction("failOpenExit", (state) => {
    if (!state.failOpenConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("failOpen", () => {
    const client = new SSMClient({});
    return client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: `fake-table-${Date.now()}`,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testFailOpen", MESSAGES.demoFailOpenTest),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "resetTableConfirmation",
    MESSAGES.demoResetTableConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("resetTableExit", (state) => {
    if (!state.resetTableConfirmation) {
      process.exit();
    }
  }),
```

```
  new ScenarioAction("resetTable", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testResetTable", MESSAGES.demoTestResetTable),
  healthCheckLoop,
  loadBalancerLoop,
];

async function createSsmOnlyInstanceProfile() {
  const iamClient = new IAMClient({});
  const { Policy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyName: NAMES.ssmOnlyPolicyName,
      PolicyDocument: readFileSync(
        join(RESOURCES_PATH, "ssm_only_policy.json"),
      ),
    }),
  );
  await iamClient.send(
    new CreateRoleCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      AssumeRolePolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
            Effect: "Allow",
            Principal: { Service: "ec2.amazonaws.com" },
            Action: "sts:AssumeRole",
          },
        ],
      }),
    }),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
```

```
      PolicyArn: Policy.Arn,
    }),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
    }),
  );
  const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
  await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
  await iamClient.send(
    new AddRoleToInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
      RoleName: NAMES.ssmOnlyRoleName,
    }),
  );

  return InstanceProfile;
}
```

Créez des étapes pour déployer toutes les ressources.

```
import { unlinkSync } from "node:fs";

import { DynamoDBClient, DeleteTableCommand } from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  DeleteKeyPairCommand,
  DeleteLaunchTemplateCommand,
  RevokeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  DeleteInstanceProfileCommand,
```

```
  RemoveRoleFromInstanceProfileCommand,
  DeletePolicyCommand,
  DeleteRoleCommand,
  DetachRolePolicyCommand,
  paginateListPolicies,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DeleteAutoScalingGroupCommand,
  TerminateInstanceInAutoScalingGroupCommand,
  UpdateAutoScalingGroupCommand,
  paginateDescribeAutoScalingGroups,
} from "@aws-sdk/client-auto-scaling";
import {
  DeleteLoadBalancerCommand,
  DeleteTargetGroupCommand,
  DescribeTargetGroupsCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { loadState } from "@aws-doc-sdk-examples/lib/scenario/steps-common.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const destroySteps = [
  loadState,
  new ScenarioInput("destroy", MESSAGES.destroy, { type: "confirm" }),
  new ScenarioAction(
    "abort",
    (state) => state.destroy === false && process.exit(),
  ),
  new ScenarioAction("deleteTable", async (c) => {
    try {
      const client = new DynamoDBClient({});
```

```
      await client.send(new DeleteTableCommand({ TableName: NAMES.tableName }));
    } catch (e) {
      c.deleteTableError = e;
    }
  }),
  new ScenarioOutput("deleteTableResult", (state) => {
    if (state.deleteTableError) {
      console.error(state.deleteTableError);
      return MESSAGES.deleteTableError.replace(
        "${TABLE_NAME}",
        NAMES.tableName,
      );
    }
    return MESSAGES.deletedTable.replace("${TABLE_NAME}", NAMES.tableName);
  }),
  new ScenarioAction("deleteKeyPair", async (state) => {
    try {
      const client = new EC2Client({});
      await client.send(
        new DeleteKeyPairCommand({ KeyName: NAMES.keyPairName }),
      );
      unlinkSync(`${NAMES.keyPairName}.pem`);
    } catch (e) {
      state.deleteKeyPairError = e;
    }
  }),
  new ScenarioOutput("deleteKeyPairResult", (state) => {
    if (state.deleteKeyPairError) {
      console.error(state.deleteKeyPairError);
      return MESSAGES.deleteKeyPairError.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
    }
    return MESSAGES.deletedKeyPair.replace(
      "${KEY_PAIR_NAME}",
      NAMES.keyPairName,
    );
  }),
  new ScenarioAction("detachPolicyFromRole", async (state) => {
    try {
      const client = new IAMClient({});
      const policy = await findPolicy(NAMES.instancePolicyName);
```

```
      if (!policy) {
        state.detachPolicyFromRoleError = new Error(
          `Policy ${NAMES.instancePolicyName} not found.`,
        );
      } else {
        await client.send(
          new DetachRolePolicyCommand({
            RoleName: NAMES.instanceRoleName,
            PolicyArn: policy.Arn,
          }),
        );
      }
    } catch (e) {
      state.detachPolicyFromRoleError = e;
    }
  }),
  new ScenarioOutput("detachedPolicyFromRole", (state) => {
    if (state.detachPolicyFromRoleError) {
      console.error(state.detachPolicyFromRoleError);
      return MESSAGES.detachPolicyFromRoleError
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
    return MESSAGES.detachedPolicyFromRole
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
  }),
  new ScenarioAction("deleteInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const policy = await findPolicy(NAMES.instancePolicyName);

    if (!policy) {
      state.deletePolicyError = new Error(
        `Policy ${NAMES.instancePolicyName} not found.`,
      );
    } else {
      return client.send(
        new DeletePolicyCommand({
          PolicyArn: policy.Arn,
        }),
      );
    }
  }),
  new ScenarioOutput("deletePolicyResult", (state) => {
```

```
    if (state.deletePolicyError) {
      console.error(state.deletePolicyError);
      return MESSAGES.deletePolicyError.replace(
        "${INSTANCE_POLICY_NAME}",
        NAMES.instancePolicyName,
      );
    }
    return MESSAGES.deletedPolicy.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    );
  }),
  new ScenarioAction("removeRoleFromInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          RoleName: NAMES.instanceRoleName,
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.removeRoleFromInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("removeRoleFromInstanceProfileResult", (state) => {
    if (state.removeRoleFromInstanceProfile) {
      console.error(state.removeRoleFromInstanceProfileError);
      return MESSAGES.removeRoleFromInstanceProfileError
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
    return MESSAGES.removedRoleFromInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
  }),
  new ScenarioAction("deleteInstanceRole", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteRoleCommand({
          RoleName: NAMES.instanceRoleName,
        }),
      );
```

```
    } catch (e) {
      state.deleteInstanceRoleError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceRoleResult", (state) => {
    if (state.deleteInstanceRoleError) {
      console.error(state.deleteInstanceRoleError);
      return MESSAGES.deleteInstanceRoleError.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    }
    return MESSAGES.deletedInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    );
  }),
  new ScenarioAction("deleteInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.deleteInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceProfileResult", (state) => {
    if (state.deleteInstanceProfileError) {
      console.error(state.deleteInstanceProfileError);
      return MESSAGES.deleteInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    }
    return MESSAGES.deletedInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    );
  }),
  new ScenarioAction("deleteAutoScalingGroup", async (state) => {
    try {
```

```
      await terminateGroupInstances(NAMES.autoScalingGroupName);
      await retry({ intervalInMs: 60000, maxRetries: 60 }, async () => {
        await deleteAutoScalingGroup(NAMES.autoScalingGroupName);
      });
    } catch (e) {
      state.deleteAutoScalingGroupError = e;
    }
  }),
  new ScenarioOutput("deleteAutoScalingGroupResult", (state) => {
    if (state.deleteAutoScalingGroupError) {
      console.error(state.deleteAutoScalingGroupError);
      return MESSAGES.deleteAutoScalingGroupError.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    }
    return MESSAGES.deletedAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    );
  }),
  new ScenarioAction("deleteLaunchTemplate", async (state) => {
    const client = new EC2Client({});
    try {
      await client.send(
        new DeleteLaunchTemplateCommand({
          LaunchTemplateName: NAMES.launchTemplateName,
        }),
      );
    } catch (e) {
      state.deleteLaunchTemplateError = e;
    }
  }),
  new ScenarioOutput("deleteLaunchTemplateResult", (state) => {
    if (state.deleteLaunchTemplateError) {
      console.error(state.deleteLaunchTemplateError);
      return MESSAGES.deleteLaunchTemplateError.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    }
    return MESSAGES.deletedLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
```

```
    );
  }),
  new ScenarioAction("deleteLoadBalancer", async (state) => {
    try {
      const client = new ElasticLoadBalancingV2Client({});
      const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
      await client.send(
        new DeleteLoadBalancerCommand({
          LoadBalancerArn: loadBalancer.LoadBalancerArn,
        }),
      );
      await retry({ intervalInMs: 1000, maxRetries: 60 }, async () => {
        const lb = await findLoadBalancer(NAMES.loadBalancerName);
        if (lb) {
          throw new Error("Load balancer still exists.");
        }
      });
    } catch (e) {
      state.deleteLoadBalancerError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerResult", (state) => {
    if (state.deleteLoadBalancerError) {
      console.error(state.deleteLoadBalancerError);
      return MESSAGES.deleteLoadBalancerError.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    }
    return MESSAGES.deletedLoadBalancer.replace(
      "${LB_NAME}",
      NAMES.loadBalancerName,
    );
  }),
  new ScenarioAction("deleteLoadBalancerTargetGroup", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    try {
      const { TargetGroups } = await client.send(
        new DescribeTargetGroupsCommand({
          Names: [NAMES.loadBalancerTargetGroupName],
        }),
      );

      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
```

```
        client.send(
          new DeleteTargetGroupCommand({
            TargetGroupArn: TargetGroups[0].TargetGroupArn,
          }),
        ),
      );
    } catch (e) {
      state.deleteLoadBalancerTargetGroupError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerTargetGroupResult", (state) => {
    if (state.deleteLoadBalancerTargetGroupError) {
      console.error(state.deleteLoadBalancerTargetGroupError);
      return MESSAGES.deleteLoadBalancerTargetGroupError.replace(
        "${TARGET_GROUP_NAME}",
        NAMES.loadBalancerTargetGroupName,
      );
    }
    return MESSAGES.deletedLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    );
  }),
  new ScenarioAction("detachSsmOnlyRoleFromProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyRoleFromProfileError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyRoleFromProfileResult", (state) => {
    if (state.detachSsmOnlyRoleFromProfileError) {
      console.error(state.detachSsmOnlyRoleFromProfileError);
      return MESSAGES.detachSsmOnlyRoleFromProfileError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    }
    return MESSAGES.detachedSsmOnlyRoleFromProfile
```

```
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
  }),
  new ScenarioAction("detachSsmOnlyCustomRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyCustomRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyCustomRolePolicyResult", (state) => {
    if (state.detachSsmOnlyCustomRolePolicyError) {
      console.error(state.detachSsmOnlyCustomRolePolicyError);
      return MESSAGES.detachSsmOnlyCustomRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    }
    return MESSAGES.detachedSsmOnlyCustomRolePolicy
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
  }),
  new ScenarioAction("detachSsmOnlyAWSRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
        }),
      );
    } catch (e) {
      state.detachSsmOnlyAWSRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyAWSRolePolicyResult", (state) => {
    if (state.detachSsmOnlyAWSRolePolicyError) {
      console.error(state.detachSsmOnlyAWSRolePolicyError);
```

```
      return MESSAGES.detachSsmOnlyAWSRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
    }
    return MESSAGES.detachedSsmOnlyAWSRolePolicy
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
  }),
  new ScenarioAction("deleteSsmOnlyInstanceProfile", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyInstanceProfileResult", (state) => {
    if (state.deleteSsmOnlyInstanceProfileError) {
      console.error(state.deleteSsmOnlyInstanceProfileError);
      return MESSAGES.deleteSsmOnlyInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    }
    return MESSAGES.deletedSsmOnlyInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.ssmOnlyInstanceProfileName,
    );
  }),
  new ScenarioAction("deleteSsmOnlyPolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DeletePolicyCommand({
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyPolicyError = e;
```

```
    }
  }),
  new ScenarioOutput("deleteSsmOnlyPolicyResult", (state) => {
    if (state.deleteSsmOnlyPolicyError) {
      console.error(state.deleteSsmOnlyPolicyError);
      return MESSAGES.deleteSsmOnlyPolicyError.replace(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    }
    return MESSAGES.deletedSsmOnlyPolicy.replace(
      "${POLICY_NAME}",
      NAMES.ssmOnlyPolicyName,
    );
  }),
  new ScenarioAction("deleteSsmOnlyRole", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteRoleCommand({
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyRoleError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyRoleResult", (state) => {
    if (state.deleteSsmOnlyRoleError) {
      console.error(state.deleteSsmOnlyRoleError);
      return MESSAGES.deleteSsmOnlyRoleError.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    }
    return MESSAGES.deletedSsmOnlyRole.replace(
      "${ROLE_NAME}",
      NAMES.ssmOnlyRoleName,
    );
  }),
  new ScenarioAction(
    "revokeSecurityGroupIngress",
    async (
```

```
      /** @type {{ myIp: string, defaultSecurityGroup: { GroupId: string } }} */
 state,
    ) => {
      const ec2Client = new EC2Client({});

      try {
        await ec2Client.send(
          new RevokeSecurityGroupIngressCommand({
            GroupId: state.defaultSecurityGroup.GroupId,
            CidrIp: `${state.myIp}/32`,
            FromPort: 80,
            ToPort: 80,
            IpProtocol: "tcp",
          }),
        );
      } catch (e) {
        state.revokeSecurityGroupIngressError = e;
      }
    },
  ),
  new ScenarioOutput("revokeSecurityGroupIngressResult", (state) => {
    if (state.revokeSecurityGroupIngressError) {
      console.error(state.revokeSecurityGroupIngressError);
      return MESSAGES.revokeSecurityGroupIngressError.replace(
        "${IP}",
        state.myIp,
      );
    }
    return MESSAGES.revokedSecurityGroupIngress.replace("${IP}", state.myIp);
  }),
];

/**
 * @param {string} policyName
 */
async function findPolicy(policyName) {
  const client = new IAMClient({});
  const paginatedPolicies = paginateListPolicies({ client }, {});
  for await (const page of paginatedPolicies) {
    const policy = page.Policies.find((p) => p.PolicyName === policyName);
    if (policy) {
      return policy;
    }
  }
```

```
}

/**
 * @param {string} groupName
 */
async function deleteAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  try {
    await client.send(
      new DeleteAutoScalingGroupCommand({
        AutoScalingGroupName: groupName,
      }),
    );
  } catch (err) {
    if (!(err instanceof Error)) {
      throw err;
    }
    console.log(err.name);
    throw err;
  }
}

/**
 * @param {string} groupName
 */
async function terminateGroupInstances(groupName) {
  const autoScalingClient = new AutoScalingClient({});
  const group = await findAutoScalingGroup(groupName);
  await autoScalingClient.send(
    new UpdateAutoScalingGroupCommand({
      AutoScalingGroupName: group.AutoScalingGroupName,
      MinSize: 0,
    }),
  );
  for (const i of group.Instances) {
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: i.InstanceId,
          ShouldDecrementDesiredCapacity: true,
        }),
      ),
    );
  }
```

```
}

async function findAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  const paginatedGroups = paginateDescribeAutoScalingGroups({ client }, {});
  for await (const page of paginatedGroups) {
    const group = page.AutoScalingGroups.find(
      (g) => g.AutoScalingGroupName === groupName,
    );
    if (group) {
      return group;
    }
  }
  throw new Error(`Auto scaling group ${groupName} not found.`);
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la référence de l'API AWS SDK pour JavaScript .
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadBalancers

- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Exécutez un scénario interactif à une invite de commande.

```
class Runner:
    """
    Manages the deployment, demonstration, and destruction of resources for the
 resilient service.
    """

    def __init__(
        self,
        resource_path: str,
        recommendation: RecommendationService,
        autoscaler: AutoScalingWrapper,
        loadbalancer: ElasticLoadBalancerWrapper,
        param_helper: ParameterHelper,
    ):
        """
        Initializes the Runner class with the necessary parameters.
```

```
        :param resource_path: The path to resource files used by this example,
such as IAM policies and instance scripts.
        :param recommendation: An instance of the RecommendationService class.
        :param autoscaler: An instance of the AutoScaler class.
        :param loadbalancer: An instance of the LoadBalancer class.
        :param param_helper: An instance of the ParameterHelper class.
        """
        self.resource_path = resource_path
        self.recommendation = recommendation
        self.autoscaler = autoscaler
        self.loadbalancer = loadbalancer
        self.param_helper = param_helper
        self.protocol = "HTTP"
        self.port = 80
        self.ssh_port = 22

        prefix = "doc-example-resilience"
        self.target_group_name = f"{prefix}-tg"
        self.load_balancer_name = f"{prefix}-lb"

    def deploy(self) -> None:
        """
        Deploys the resources required for the resilient service, including the
DynamoDB table,
        EC2 instances, Auto Scaling group, and load balancer.
        """
        recommendations_path = f"{self.resource_path}/recommendations.json"
        startup_script = f"{self.resource_path}/server_startup_script.sh"
        instance_policy = f"{self.resource_path}/instance_policy.json"

        logging.info("Starting deployment of resources for the resilient
service.")

        logging.info(
            "Creating and populating DynamoDB table '%s'.",
            self.recommendation.table_name,
        )
        self.recommendation.create()
        self.recommendation.populate(recommendations_path)

        logging.info(
            "Creating an EC2 launch template with the startup script '%s'.",
            startup_script,
        )
```

```
        self.autoscaler.create_template(startup_script, instance_policy)

        logging.info(
            "Creating an EC2 Auto Scaling group across multiple Availability
 Zones."
        )
        zones = self.autoscaler.create_autoscaling_group(3)

        logging.info("Creating variables that control the flow of the demo.")
        self.param_helper.reset()

        logging.info("Creating Elastic Load Balancing target group and load
 balancer.")

        vpc = self.autoscaler.get_default_vpc()
        subnets = self.autoscaler.get_subnets(vpc["VpcId"], zones)
        target_group = self.loadbalancer.create_target_group(
            self.target_group_name, self.protocol, self.port, vpc["VpcId"]
        )
        self.loadbalancer.create_load_balancer(
            self.load_balancer_name, [subnet["SubnetId"] for subnet in subnets]
        )
        self.loadbalancer.create_listener(self.load_balancer_name, target_group)

        self.autoscaler.attach_load_balancer_target_group(target_group)

        logging.info("Verifying access to the load balancer endpoint.")
        endpoint = self.loadbalancer.get_endpoint(self.load_balancer_name)
        lb_success = self.loadbalancer.verify_load_balancer_endpoint(endpoint)
        current_ip_address = requests.get("http://
checkip.amazonaws.com").text.strip()

        if not lb_success:
            logging.warning(
                "Couldn't connect to the load balancer. Verifying that the port
 is open..."
            )
            sec_group, port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.port, current_ip_address
            )
            sec_group, ssh_port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.ssh_port, current_ip_address
            )
            if not port_is_open:
```

```
                logging.warning(
                    "The default security group for your VPC must allow access
from this computer."
                )
                if q.ask(
                    f"Do you want to add a rule to security group
{sec_group['GroupId']} to allow\n"
                    f"inbound traffic on port {self.port} from your computer's IP
address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.port, current_ip_address
                    )
            if not ssh_port_is_open:
                if q.ask(
                    f"Do you want to add a rule to security group
{sec_group['GroupId']} to allow\n"
                    f"inbound SSH traffic on port {self.ssh_port} for debugging
from your computer's IP address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.ssh_port, current_ip_address
                    )
            lb_success =
self.loadbalancer.verify_load_balancer_endpoint(endpoint)

        if lb_success:
            logging.info(
                "Load balancer is ready. Access it at: http://%s",
current_ip_address
            )
        else:
            logging.error(
                "Couldn't get a successful response from the load balancer
endpoint. Please verify your VPC and security group settings."
            )

    def demo_choices(self) -> None:
        """
        Presents choices for interacting with the deployed service, such as
sending requests to
        the load balancer or checking the health of the targets.
```

```
        """
        actions = [
            "Send a GET request to the load balancer endpoint.",
            "Check the health of load balancer targets.",
            "Go to the next part of the demo.",
        ]
        choice = 0
        while choice != 2:
            logging.info("Choose an action to interact with the service.")
            choice = q.choose("Which action would you like to take? ", actions)
            if choice == 0:
                logging.info("Sending a GET request to the load balancer
endpoint.")
                endpoint =
self.loadbalancer.get_endpoint(self.load_balancer_name)
                logging.info("GET http://%s", endpoint)
                response = requests.get(f"http://{endpoint}")
                logging.info("Response: %s", response.status_code)
                if response.headers.get("content-type") == "application/json":
                    pp(response.json())
            elif choice == 1:
                logging.info("Checking the health of load balancer targets.")
                health =
self.loadbalancer.check_target_health(self.target_group_name)
                for target in health:
                    state = target["TargetHealth"]["State"]
                    logging.info(
                        "Target %s on port %d is %s",
                        target["Target"]["Id"],
                        target["Target"]["Port"],
                        state,
                    )
                    if state != "healthy":
                        logging.warning(
                            "%s: %s",
                            target["TargetHealth"]["Reason"],
                            target["TargetHealth"]["Description"],
                        )
                logging.info(
                    "Note that it can take a minute or two for the health check
to update."
                )
            elif choice == 2:
                logging.info("Proceeding to the next part of the demo.")
```

```
   def demo(self) -> None:
       """
       Runs the demonstration, showing how the service responds to different
failure scenarios
       and how a resilient architecture can keep the service running.
       """
       ssm_only_policy = f"{self.resource_path}/ssm_only_policy.json"

       logging.info("Resetting parameters to starting values for the demo.")
       self.param_helper.reset()

       logging.info(
           "Starting demonstration of the service's resilience under various
failure conditions."
       )
       self.demo_choices()

       logging.info(
           "Simulating failure by changing the Systems Manager parameter to a
non-existent table."
       )
       self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
       logging.info("Sending GET requests will now return failure codes.")
       self.demo_choices()

       logging.info("Switching to static response mode to mitigate failure.")
       self.param_helper.put(self.param_helper.failure_response, "static")
       logging.info("Sending GET requests will now return static responses.")
       self.demo_choices()

       logging.info("Restoring normal operation of the recommendation service.")
       self.param_helper.put(self.param_helper.table,
self.recommendation.table_name)

       logging.info(
           "Introducing a failure by assigning bad credentials to one of the
instances."
       )
       self.autoscaler.create_instance_profile(
           ssm_only_policy,
           self.autoscaler.bad_creds_policy_name,
           self.autoscaler.bad_creds_role_name,
           self.autoscaler.bad_creds_profile_name,
```

```
            ["AmazonSSMManagedInstanceCore"],
        )
        instances = self.autoscaler.get_instances()
        bad_instance_id = instances[0]
        instance_profile = self.autoscaler.get_instance_profile(bad_instance_id)
        logging.info(
            "Replacing instance profile with bad credentials for instance %s.",
            bad_instance_id,
        )
        self.autoscaler.replace_instance_profile(
            bad_instance_id,
            self.autoscaler.bad_creds_profile_name,
            instance_profile["AssociationId"],
        )
        logging.info(
            "Sending GET requests may return either a valid recommendation or a
static response."
        )
        self.demo_choices()

        logging.info("Implementing deep health checks to detect unhealthy
instances.")
        self.param_helper.put(self.param_helper.health_check, "deep")
        logging.info("Checking the health of the load balancer targets.")
        self.demo_choices()

        logging.info(
            "Terminating the unhealthy instance to let the auto scaler replace
it."
        )
        self.autoscaler.terminate_instance(bad_instance_id)
        logging.info("The service remains resilient during instance
replacement.")
        self.demo_choices()

        logging.info("Simulating a complete failure of the recommendation
service.")
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        logging.info(
            "All instances will report as unhealthy, but the service will still
return static responses."
        )
        self.demo_choices()
        self.param_helper.reset()
```

```
    def destroy(self, automation=False) -> None:
        """
        Destroys all resources created for the demo, including the load balancer,
 Auto Scaling group,
        EC2 instances, and DynamoDB table.
        """
        logging.info(
            "This concludes the demo. Preparing to clean up all AWS resources
 created during the demo."
        )
        if automation:
            cleanup = True
        else:
            cleanup = q.ask(
                "Do you want to clean up all demo resources? (y/n) ", q.is_yesno
            )

        if cleanup:
            logging.info("Deleting load balancer and related resources.")
            self.loadbalancer.delete_load_balancer(self.load_balancer_name)
            self.loadbalancer.delete_target_group(self.target_group_name)
            self.autoscaler.delete_autoscaling_group(self.autoscaler.group_name)
            self.autoscaler.delete_key_pair()
            self.autoscaler.delete_template()
            self.autoscaler.delete_instance_profile(
                self.autoscaler.bad_creds_profile_name,
                self.autoscaler.bad_creds_role_name,
            )
            logging.info("Deleting DynamoDB table and other resources.")
            self.recommendation.destroy()
        else:
            logging.warning(
                "Resources have not been deleted. Ensure you clean them up
 manually to avoid unexpected charges."
            )


def main() -> None:
    """
    Main function to parse arguments and run the appropriate actions for the
 demo.
    """
    parser = argparse.ArgumentParser()
```

```
    parser.add_argument(
        "--action",
        required=True,
        choices=["all", "deploy", "demo", "destroy"],
        help="The action to take for the demo. When 'all' is specified, resources
are\n"
        "deployed, the demo is run, and resources are destroyed.",
    )
    parser.add_argument(
        "--resource_path",
        default="../../../scenarios/features/resilient_service/resources",
        help="The path to resource files used by this example, such as IAM
policies and\n"
        "instance scripts.",
    )
    args = parser.parse_args()

    logging.info("Starting the Resilient Service demo.")

    prefix = "doc-example-resilience"

    # Service Clients
    ddb_client = boto3.client("dynamodb")
    elb_client = boto3.client("elbv2")
    autoscaling_client = boto3.client("autoscaling")
    ec2_client = boto3.client("ec2")
    ssm_client = boto3.client("ssm")
    iam_client = boto3.client("iam")

    # Wrapper instantiations
    recommendation = RecommendationService(
        "doc-example-recommendation-service", ddb_client
    )
    autoscaling_wrapper = AutoScalingWrapper(
        prefix,
        "t3.micro",
        "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    )
    elb_wrapper = ElasticLoadBalancerWrapper(elb_client)
    param_helper = ParameterHelper(recommendation.table_name, ssm_client)
```

```
    # Demo invocation
    runner = Runner(
        args.resource_path,
        recommendation,
        autoscaling_wrapper,
        elb_wrapper,
        param_helper,
    )
    actions = [args.action] if args.action != "all" else ["deploy", "demo",
 "destroy"]
    for action in actions:
        if action == "deploy":
            runner.deploy()
        elif action == "demo":
            runner.demo()
        elif action == "destroy":
            runner.destroy()

    logging.info("Demo completed successfully.")


if __name__ == "__main__":
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    main()
```

Créez une classe qui englobe les actions Auto Scaling et Amazon EC2.

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
```

```
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def create_policy(self, policy_file: str, policy_name: str) -> str:
        """
        Creates a new IAM policy or retrieves the ARN of an existing policy.

        :param policy_file: The path to a JSON file that contains the policy
definition.
```

```
        :param policy_name: The name to give the created policy.
        :return: The ARN of the created or existing policy.
        """
        with open(policy_file) as file:
            policy_doc = file.read()

        try:
            response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=policy_doc
            )
            policy_arn = response["Policy"]["Arn"]
            log.info(f"Policy '{policy_name}' created successfully. ARN:
 {policy_arn}")
            return policy_arn

        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                # If the policy already exists, get its ARN
                response = self.iam_client.get_policy(
                    PolicyArn=f"arn:aws:iam::{self.account_id}:policy/
{policy_name}"
                )
                policy_arn = response["Policy"]["Arn"]
                log.info(f"Policy '{policy_name}' already exists. ARN:
 {policy_arn}")
                return policy_arn
            log.error(f"Full error:\n\t{err}")

    def create_role(self, role_name: str, assume_role_doc: dict) -> str:
        """
        Creates a new IAM role or retrieves the ARN of an existing role.

        :param role_name: The name to give the created role.
        :param assume_role_doc: The assume role policy document that specifies
 which
                                entities can assume the role.
        :return: The ARN of the created or existing role.
        """
        try:
            response = self.iam_client.create_role(
                RoleName=role_name,
 AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            role_arn = response["Role"]["Arn"]
```

```
            log.info(f"Role '{role_name}' created successfully. ARN: {role_arn}")
            return role_arn

        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                # If the role already exists, get its ARN
                response = self.iam_client.get_role(RoleName=role_name)
                role_arn = response["Role"]["Arn"]
                log.info(f"Role '{role_name}' already exists. ARN: {role_arn}")
                return role_arn
            log.error(f"Full error:\n\t{err}")

    def attach_policy(
        self,
        role_name: str,
        policy_arn: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> None:
        """
        Attaches an IAM policy to a role and optionally attaches additional AWS-
managed policies.

        :param role_name: The name of the role to attach the policy to.
        :param policy_arn: The ARN of the policy to attach.
        :param aws_managed_policies: A tuple of AWS-managed policy names to
 attach to the role.
        """
        try:
            self.iam_client.attach_role_policy(RoleName=role_name,
 PolicyArn=policy_arn)
            for aws_policy in aws_managed_policies:
                self.iam_client.attach_role_policy(
                    RoleName=role_name,
                    PolicyArn=f"arn:aws:iam::aws:policy/{aws_policy}",
                )
            log.info(f"Attached policy {policy_arn} to role {role_name}.")
        except ClientError as err:
            log.error(f"Failed to attach policy {policy_arn} to role
 {role_name}.")
            log.error(f"Full error:\n\t{err}")

    def create_instance_profile(
        self,
        policy_file: str,
```

```
        policy_name: str,
        role_name: str,
        profile_name: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> str:
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        policy_arn = self.create_policy(policy_file, policy_name)
        self.create_role(role_name, assume_role_doc)
        self.attach_policy(role_name, policy_arn, aws_managed_policies)

        try:
            profile_response = self.iam_client.create_instance_profile(
```

```
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            log.error(f"Full error:\n\t{err}")
        return profile_arn


    def get_instance_profile(self, instance_id: str) -> Dict[str, Any]:
        """
        Gets data about the profile associated with an instance.

        :param instance_id: The ID of the instance to look up.
        :return: The profile data.
        """
        try:
            response =
self.ec2_client.describe_iam_instance_profile_associations(
                Filters=[{"Name": "instance-id", "Values": [instance_id]}]
            )
            if not response["IamInstanceProfileAssociations"]:
                log.info(f"No instance profile found for instance
{instance_id}.")
            profile_data = response["IamInstanceProfileAssociations"][0]
            log.info(f"Retrieved instance profile for instance {instance_id}.")
            return profile_data
        except ClientError as err:
```

```
            log.error(
                f"Failed to retrieve instance profile for instance
{instance_id}."
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidInstanceID.NotFound":
                log.error(f"The instance ID '{instance_id}' does not exist.")
            log.error(f"Full error:\n\t{err}")


    def replace_instance_profile(
        self,
        instance_id: str,
        new_instance_profile_name: str,
        profile_association_id: str,
    ) -> None:
        """
        Replaces the profile associated with a running instance. After the
profile is
        replaced, the instance is rebooted to ensure that it uses the new
profile. When
        the instance is ready, Systems Manager is used to restart the Python web
server.

        :param instance_id: The ID of the instance to restart.
        :param new_instance_profile_name: The name of the new profile to
associate with
                                          the specified instance.
        :param profile_association_id: The ID of the existing profile association
for the
                                       instance.
        """
        try:
            self.ec2_client.replace_iam_instance_profile_association(
                IamInstanceProfile={"Name": new_instance_profile_name},
                AssociationId=profile_association_id,
            )
            log.info(
                "Replaced instance profile for association %s with profile %s.",
                profile_association_id,
                new_instance_profile_name,
            )
            time.sleep(5)
```

```
            self.ec2_client.reboot_instances(InstanceIds=[instance_id])
            log.info("Rebooting instance %s.", instance_id)
            waiter = self.ec2_client.get_waiter("instance_running")
            log.info("Waiting for instance %s to be running.", instance_id)
            waiter.wait(InstanceIds=[instance_id])
            log.info("Instance %s is now running.", instance_id)

            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info(f"Restarted the Python web server on instance
'{instance_id}'.")
        except ClientError as err:
            log.error("Failed to replace instance profile.")
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidAssociationID.NotFound":
                log.error(
                    f"Association ID '{profile_association_id}' does not exist."
                    "Please check the association ID and try again."
                )
            if error_code == "InvalidInstanceId":
                log.error(
                    f"The specified instance ID '{instance_id}' does not exist or
is not available for SSM. "
                    f"Please verify the instance ID and try again."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_instance_profile(self, profile_name: str, role_name: str) -> None:
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
```

```
self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            log.error(
                f"Couldn't delete instance profile {profile_name} or detach "
                f"policies and delete role {role_name}: {err}"
            )
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )


    def create_key_pair(self, key_pair_name: str) -> None:
        """
        Creates a new key pair.

        :param key_pair_name: The name of the key pair to create.
        """
        try:
            response = self.ec2_client.create_key_pair(KeyName=key_pair_name)
            with open(f"{key_pair_name}.pem", "w") as file:
                file.write(response["KeyMaterial"])
            chmod(f"{key_pair_name}.pem", 0o600)
            log.info("Created key pair %s.", key_pair_name)
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to create key pair {key_pair_name}.")
            if error_code == "InvalidKeyPair.Duplicate":
```

```
                log.error(f"A key pair with the name '{key_pair_name}' already
exists.")
            log.error(f"Full error:\n\t{err}")


    def delete_key_pair(self) -> None:
        """
        Deletes a key pair.
        """
        try:
            self.ec2_client.delete_key_pair(KeyName=self.key_pair_name)
            remove(f"{self.key_pair_name}.pem")
            log.info("Deleted key pair %s.", self.key_pair_name)
        except ClientError as err:
            log.error(f"Couldn't delete key pair '{self.key_pair_name}'.")
            log.error(f"Full error:\n\t{err}")
        except FileNotFoundError as err:
            log.info("Key pair %s doesn't exist, nothing to do.",
self.key_pair_name)
            log.error(f"Full error:\n\t{err}")


    def create_template(
        self, server_startup_script_file: str, instance_policy_file: str
    ) -> Dict[str, Any]:
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling. The
        launch template specifies a Bash script in its user data field that runs
after
        the instance is started. This script installs Python packages and starts
a
        Python web server on the instance.

        :param server_startup_script_file: The path to a Bash script file that is
run
                                           when an instance starts.
        :param instance_policy_file: The path to a file that defines a
permissions policy
                                     to create and attach to the instance
profile.
        :return: Information about the newly created template.
        """
        template = {}
```

```
        try:
            # Create key pair and instance profile
            self.create_key_pair(self.key_pair_name)
            self.create_instance_profile(
                instance_policy_file,
                self.instance_policy_name,
                self.instance_role_name,
                self.instance_profile_name,
            )

            # Read the startup script
            with open(server_startup_script_file) as file:
                start_server_script = file.read()

            # Get the latest AMI ID
            ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
            ami_id = ami_latest["Parameter"]["Value"]

            # Create the launch template
            lt_response = self.ec2_client.create_launch_template(
                LaunchTemplateName=self.launch_template_name,
                LaunchTemplateData={
                    "InstanceType": self.inst_type,
                    "ImageId": ami_id,
                    "IamInstanceProfile": {"Name": self.instance_profile_name},
                    "UserData": base64.b64encode(
                        start_server_script.encode(encoding="utf-8")
                    ).decode(encoding="utf-8"),
                    "KeyName": self.key_pair_name,
                },
            )
            template = lt_response["LaunchTemplate"]
            log.info(
                f"Created launch template {self.launch_template_name} for AMI
{ami_id} on {self.inst_type}."
            )
        except ClientError as err:
            log.error(f"Failed to create launch template
{self.launch_template_name}.")
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidLaunchTemplateName.AlreadyExistsException":
                log.info(
                    f"Launch template {self.launch_template_name} already exists,
nothing to do."
```

```
                )
            log.error(f"Full error:\n\t{err}")
        return template


    def delete_template(self):
        """
        Deletes a launch template.
        """
        try:
            self.ec2_client.delete_launch_template(
                LaunchTemplateName=self.launch_template_name
            )
            self.delete_instance_profile(
                self.instance_profile_name, self.instance_role_name
            )
            log.info("Launch template %s deleted.", self.launch_template_name)
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.NotFoundException"
            ):
                log.info(
                    "Launch template %s does not exist, nothing to do.",
                    self.launch_template_name,
                )
            log.error(f"Full error:\n\t{err}")


    def get_availability_zones(self) -> List[str]:
        """
        Gets a list of Availability Zones in the AWS Region of the Amazon EC2
client.

        :return: The list of Availability Zones for the client Region.
        """
        try:
            response = self.ec2_client.describe_availability_zones()
            zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
            log.info(f"Retrieved {len(zones)} availability zones: {zones}.")
        except ClientError as err:
            log.error("Failed to retrieve availability zones.")
            log.error(f"Full error:\n\t{err}")
        else:
```

```
            return zones


    def create_autoscaling_group(self, group_size: int) -> List[str]:
        """
        Creates an EC2 Auto Scaling group with the specified size.

        :param group_size: The number of instances to set for the minimum and
maximum in
                           the group.
        :return: The list of Availability Zones specified for the group.
        """
        try:
            zones = self.get_availability_zones()
            self.autoscaling_client.create_auto_scaling_group(
                AutoScalingGroupName=self.group_name,
                AvailabilityZones=zones,
                LaunchTemplate={
                    "LaunchTemplateName": self.launch_template_name,
                    "Version": "$Default",
                },
                MinSize=group_size,
                MaxSize=group_size,
            )
            log.info(
                f"Created EC2 Auto Scaling group {self.group_name} with
availability zones {zones}."
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            if error_code == "AlreadyExists":
                log.info(
                    f"EC2 Auto Scaling group {self.group_name} already exists,
nothing to do."
                )
            else:
                log.error(f"Failed to create EC2 Auto Scaling group
{self.group_name}.")
                log.error(f"Full error:\n\t{err}")
        else:
            return zones


    def get_instances(self) -> List[str]:
```

```
        """
        Gets data about the instances in the EC2 Auto Scaling group.

        :return: A list of instance IDs in the Auto Scaling group.
        """
        try:
            as_response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            instance_ids = [
                i["InstanceId"]
                for i in as_response["AutoScalingGroups"][0]["Instances"]
            ]
            log.info(
                f"Retrieved {len(instance_ids)} instances for Auto Scaling group
{self.group_name}."
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to retrieve instances for Auto Scaling group
{self.group_name}."
            )
            if error_code == "ResourceNotFound":
                log.error(f"The Auto Scaling group '{self.group_name}' does not
exist.")
            log.error(f"Full error:\n\t{err}")
        else:
            return instance_ids


    def terminate_instance(self, instance_id: str, decrementsetting=False) ->
None:
        """
        Terminates an instance in an EC2 Auto Scaling group. After an instance is
        terminated, it can no longer be accessed.

        :param instance_id: The ID of the instance to terminate.
        :param decrementsetting: If True, do not replace terminated instances.
        """
        try:
            self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                InstanceId=instance_id,
                ShouldDecrementDesiredCapacity=decrementsetting,
```

```
            )
            log.info("Terminated instance %s.", instance_id)

            # Adding a waiter to ensure the instance is terminated
            waiter = self.ec2_client.get_waiter("instance_terminated")
            log.info("Waiting for instance %s to be terminated...", instance_id)
            waiter.wait(InstanceIds=[instance_id])
            log.info(
                f"Instance '{instance_id}' has been terminated and will be
replaced."
            )

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to terminate instance '{instance_id}'.")
            if error_code == "ScalingActivityInProgressFault":
                log.error(
                    "Scaling activity is currently in progress. "
                    "Wait for the scaling activity to complete before attempting
to terminate the instance again."
                )
            elif error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
the resource."
                )
            log.error(f"Full error:\n\t{err}")

    def attach_load_balancer_target_group(
        self, lb_target_group: Dict[str, Any]
    ) -> None:
        """
        Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
Scaling group.
        The target group specifies how the load balancer forwards requests to the
instances
        in the group.

        :param lb_target_group: Data about the ELB target group to attach.
        """
        try:
            self.autoscaling_client.attach_load_balancer_target_groups(
                AutoScalingGroupName=self.group_name,
```

```
                TargetGroupARNs=[lb_target_group["TargetGroupArn"]],
            )
            log.info(
                "Attached load balancer target group %s to auto scaling group
%s.",
                lb_target_group["TargetGroupName"],
                self.group_name,
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to attach load balancer target group
'{lb_target_group['TargetGroupName']}'."
            )
            if error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
the resource."
                )
            elif error_code == "ServiceLinkedRoleFailure":
                log.error(
                    "The operation failed because the service-linked role is not
ready or does not exist. "
                    "Check that the service-linked role exists and is correctly
configured."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_autoscaling_group(self, group_name: str) -> None:
        """
        Terminates all instances in the group, then deletes the EC2 Auto Scaling
group.

        :param group_name: The name of the group to delete.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[group_name]
            )
            groups = response.get("AutoScalingGroups", [])
            if len(groups) > 0:
                self.autoscaling_client.update_auto_scaling_group(
```

```
                    AutoScalingGroupName=group_name, MinSize=0
                )
                instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
                for inst_id in instance_ids:
                    self.terminate_instance(inst_id)

                # Wait for all instances to be terminated
                if instance_ids:
                    waiter = self.ec2_client.get_waiter("instance_terminated")
                    log.info("Waiting for all instances to be terminated...")
                    waiter.wait(InstanceIds=instance_ids)
                    log.info("All instances have been terminated.")
            else:
                log.info(f"No groups found named '{group_name}'! Nothing to do.")
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to delete Auto Scaling group '{group_name}'.")
            if error_code == "ScalingActivityInProgressFault":
                log.error(
                    "Scaling activity is currently in progress. "
                    "Wait for the scaling activity to complete before attempting
 to delete the group again."
                )
            elif error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
 the group."
                )
            log.error(f"Full error:\n\t{err}")


    def get_default_vpc(self) -> Dict[str, Any]:
        """
        Gets the default VPC for the account.

        :return: Data about the default VPC.
        """
        try:
            response = self.ec2_client.describe_vpcs(
                Filters=[{"Name": "is-default", "Values": ["true"]}]
            )
        except ClientError as err:
```

```
            error_code = err.response["Error"]["Code"]
            log.error("Failed to retrieve the default VPC.")
            if error_code == "UnauthorizedOperation":
                log.error(
                    "You do not have the necessary permissions to describe VPCs.
 "
                    "Ensure that your AWS IAM user or role has the correct
 permissions."
                )
            elif error_code == "InvalidParameterValue":
                log.error(
                    "One or more parameters are invalid. Check the request
 parameters."
                )

            log.error(f"Full error:\n\t{err}")
        else:
            if "Vpcs" in response and response["Vpcs"]:
                log.info(f"Retrieved default VPC: {response['Vpcs'][0]
['VpcId']}")
                return response["Vpcs"][0]
            else:
                pass


    def verify_inbound_port(
        self, vpc: Dict[str, Any], port: int, ip_address: str
    ) -> Tuple[Dict[str, Any], bool]:
        """
        Verify the default security group of the specified VPC allows ingress
 from this
        computer. This can be done by allowing ingress from this computer's IP
        address. In some situations, such as connecting from a corporate network,
 you
        must instead specify a prefix list ID. You can also temporarily open the
 port to
        any IP address while running this example. If you do, be sure to remove
 public
        access when you're done.

        :param vpc: The VPC used by this example.
        :param port: The port to verify.
        :param ip_address: This computer's IP address.
```

```
        :return: The default security group of the specified VPC, and a value
that indicates
                 whether the specified port is open.
        """
        try:
            response = self.ec2_client.describe_security_groups(
                Filters=[
                    {"Name": "group-name", "Values": ["default"]},
                    {"Name": "vpc-id", "Values": [vpc["VpcId"]]},
                ]
            )
            sec_group = response["SecurityGroups"][0]
            port_is_open = False
            log.info(f"Found default security group {sec_group['GroupId']}.")

            for ip_perm in sec_group["IpPermissions"]:
                if ip_perm.get("FromPort", 0) == port:
                    log.info(f"Found inbound rule: {ip_perm}")
                    for ip_range in ip_perm["IpRanges"]:
                        cidr = ip_range.get("CidrIp", "")
                        if cidr.startswith(ip_address) or cidr == "0.0.0.0/0":
                            port_is_open = True
                    if ip_perm["PrefixListIds"]:
                        port_is_open = True
                    if not port_is_open:
                        log.info(
                            f"The inbound rule does not appear to be open to
either this computer's IP "
                            f"address of {ip_address}, to all IP addresses
(0.0.0.0/0), or to a prefix list ID."
                        )
                    else:
                        break
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to verify inbound rule for port {port} for VPC
{vpc['VpcId']}."
            )
            if error_code == "InvalidVpcID.NotFound":
                log.error(
                    f"The specified VPC ID '{vpc['VpcId']}' does not exist.
Please check the VPC ID."
                )
```

```
            log.error(f"Full error:\n\t{err}")
        else:
            return sec_group, port_is_open


    def open_inbound_port(self, sec_group_id: str, port: int, ip_address: str) ->
None:
        """
        Add an ingress rule to the specified security group that allows access on
the
        specified port from the specified IP address.

        :param sec_group_id: The ID of the security group to modify.
        :param port: The port to open.
        :param ip_address: The IP address that is granted access.
        """
        try:
            self.ec2_client.authorize_security_group_ingress(
                GroupId=sec_group_id,
                CidrIp=f"{ip_address}/32",
                FromPort=port,
                ToPort=port,
                IpProtocol="tcp",
            )
            log.info(
                "Authorized ingress to %s on port %s from %s.",
                sec_group_id,
                port,
                ip_address,
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to authorize ingress to security group '{sec_group_id}'
on port {port} from {ip_address}."
            )
            if error_code == "InvalidGroupId.Malformed":
                log.error(
                    "The security group ID is malformed. "
                    "Please verify that the security group ID is correct."
                )
            elif error_code == "InvalidPermission.Duplicate":
                log.error(
                    "The specified rule already exists in the security group. "
```

```
                    "Check the existing rules for this security group."
                )
            log.error(f"Full error:\n\t{err}")


    def get_subnets(self, vpc_id: str, zones: List[str] = None) -> List[Dict[str,
Any]]:
        """
        Gets the default subnets in a VPC for a specified list of Availability
Zones.

        :param vpc_id: The ID of the VPC to look up.
        :param zones: The list of Availability Zones to look up.
        :return: The list of subnets found.
        """
        # Ensure that 'zones' is a list, even if None is passed
        if zones is None:
            zones = []
        try:
            paginator = self.ec2_client.get_paginator("describe_subnets")
            page_iterator = paginator.paginate(
                Filters=[
                    {"Name": "vpc-id", "Values": [vpc_id]},
                    {"Name": "availability-zone", "Values": zones},
                    {"Name": "default-for-az", "Values": ["true"]},
                ]
            )

            subnets = []
            for page in page_iterator:
                subnets.extend(page["Subnets"])

            log.info("Found %s subnets for the specified zones.", len(subnets))
            return subnets
        except ClientError as err:
            log.error(
                f"Failed to retrieve subnets for VPC '{vpc_id}' in zones
{zones}."
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidVpcID.NotFound":
                log.error(
                    "The specified VPC ID does not exist. "
                    "Please check the VPC ID and try again."
```

```
                )
            # Add more error-specific handling as needed
            log.error(f"Full error:\n\t{err}")
```

Créez une classe qui englobe les actions Elastic Load Balancing.

```
class ElasticLoadBalancerWrapper:
    """Encapsulates Elastic Load Balancing (ELB) actions."""

    def __init__(self, elb_client: boto3.client):
        """
        Initializes the LoadBalancer class with the necessary parameters.
        """
        self.elb_client = elb_client


    def create_target_group(
        self, target_group_name: str, protocol: str, port: int, vpc_id: str
    ) -> Dict[str, Any]:
        """
        Creates an Elastic Load Balancing target group. The target group
 specifies how
        the load balancer forwards requests to instances in the group and how
 instance
        health is checked.

        To speed up this demo, the health check is configured with shortened
 times and
        lower thresholds. In production, you might want to decrease the
 sensitivity of
        your health checks to avoid unwanted failures.

        :param target_group_name: The name of the target group to create.
        :param protocol: The protocol to use to forward requests, such as 'HTTP'.
        :param port: The port to use to forward requests, such as 80.
        :param vpc_id: The ID of the VPC in which the load balancer exists.
        :return: Data about the newly created target group.
        """
        try:
```

```
            response = self.elb_client.create_target_group(
                Name=target_group_name,
                Protocol=protocol,
                Port=port,
                HealthCheckPath="/healthcheck",
                HealthCheckIntervalSeconds=10,
                HealthCheckTimeoutSeconds=5,
                HealthyThresholdCount=2,
                UnhealthyThresholdCount=2,
                VpcId=vpc_id,
            )
            target_group = response["TargetGroups"][0]
            log.info(f"Created load balancing target group
'{target_group_name}'.")
            return target_group
        except ClientError as err:
            log.error(
                f"Couldn't create load balancing target group
'{target_group_name}'."
            )
            error_code = err.response["Error"]["Code"]

            if error_code == "DuplicateTargetGroupName":
                log.error(
                    f"Target group name {target_group_name} already exists. "
                    "Check if the target group already exists."
                    "Consider using a different name or deleting the existing
target group if appropriate."
                )
            elif error_code == "TooManyTargetGroups":
                log.error(
                    "Too many target groups exist in the account. "
                    "Consider deleting unused target groups to create space for
new ones."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_target_group(self, target_group_name) -> None:
        """
        Deletes the target group.
        """
        try:
            # Describe the target group to get its ARN
```

```
            response =
self.elb_client.describe_target_groups(Names=[target_group_name])
            tg_arn = response["TargetGroups"][0]["TargetGroupArn"]

            # Delete the target group
            self.elb_client.delete_target_group(TargetGroupArn=tg_arn)
            log.info("Deleted load balancing target group %s.",
target_group_name)

            # Use a custom waiter to wait until the target group is no longer
available
            self.wait_for_target_group_deletion(self.elb_client, tg_arn)
            log.info("Target group %s successfully deleted.", target_group_name)

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to delete target group '{target_group_name}'.")
            if error_code == "TargetGroupNotFound":
                log.error(
                    "Load balancer target group either already deleted or never
existed. "
                    "Verify the name and check that the resource exists in the
AWS Console."
                )
            elif error_code == "ResourceInUseException":
                log.error(
                    "Target group still in use by another resource. "
                    "Ensure that the target group is no longer associated with
any load balancers or resources.",
                )
            log.error(f"Full error:\n\t{err}")

    def wait_for_target_group_deletion(
        self, elb_client, target_group_arn, max_attempts=10, delay=30
    ):
        for attempt in range(max_attempts):
            try:

elb_client.describe_target_groups(TargetGroupArns=[target_group_arn])
                print(
                    f"Attempt {attempt + 1}: Target group {target_group_arn}
still exists."
                )
            except ClientError as e:
```

```
                if e.response["Error"]["Code"] == "TargetGroupNotFound":
                    print(
                        f"Target group {target_group_arn} has been successfully
deleted."
                    )
                    return
                else:
                    raise
            time.sleep(delay)
        raise TimeoutError(
            f"Target group {target_group_arn} was not deleted after {max_attempts
* delay} seconds."
        )


    def create_load_balancer(
        self,
        load_balancer_name: str,
        subnet_ids: List[str],
    ) -> Dict[str, Any]:
        """
        Creates an Elastic Load Balancing load balancer that uses the specified
subnets
        and forwards requests to the specified target group.

        :param load_balancer_name: The name of the load balancer to create.
        :param subnet_ids: A list of subnets to associate with the load balancer.
        :return: Data about the newly created load balancer.
        """
        try:
            response = self.elb_client.create_load_balancer(
                Name=load_balancer_name, Subnets=subnet_ids
            )
            load_balancer = response["LoadBalancers"][0]
            log.info(f"Created load balancer '{load_balancer_name}'.")

            waiter = self.elb_client.get_waiter("load_balancer_available")
            log.info(
                f"Waiting for load balancer '{load_balancer_name}' to be
available..."
            )
            waiter.wait(Names=[load_balancer_name])
            log.info(f"Load balancer '{load_balancer_name}' is now available!")
```

```
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to create load balancer '{load_balancer_name}'. Error
code: {error_code}, Message: {err.response['Error']['Message']}"
            )

            if error_code == "DuplicateLoadBalancerNameException":
                log.error(
                    f"A load balancer with the name '{load_balancer_name}'
already exists. "
                    "Load balancer names must be unique within the AWS region. "
                    "Please choose a different name and try again."
                )
            if error_code == "TooManyLoadBalancersException":
                log.error(
                    "The maximum number of load balancers has been reached in
this account and region. "
                    "You can delete unused load balancers or request an increase
in the service quota from AWS Support."
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return load_balancer


    def create_listener(
        self,
        load_balancer_name: str,
        target_group: Dict[str, Any],
    ) -> Dict[str, Any]:
        """
        Creates a listener for the specified load balancer that forwards requests
to the
        specified target group.

        :param load_balancer_name: The name of the load balancer to create a
listener for.
        :param target_group: An existing target group that is added as a listener
to the
                             load balancer.
        :return: Data about the newly created listener.
        """
        try:
```

```
            # Retrieve the load balancer ARN
            load_balancer_response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            load_balancer_arn = load_balancer_response["LoadBalancers"][0][
                "LoadBalancerArn"
            ]

            # Create the listener
            response = self.elb_client.create_listener(
                LoadBalancerArn=load_balancer_arn,
                Protocol=target_group["Protocol"],
                Port=target_group["Port"],
                DefaultActions=[
                    {
                        "Type": "forward",
                        "TargetGroupArn": target_group["TargetGroupArn"],
                    }
                ],
            )
            log.info(
                f"Created listener to forward traffic from load balancer
'{load_balancer_name}' to target group '{target_group['TargetGroupName']}'."
            )
            return response["Listeners"][0]
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to add a listener on '{load_balancer_name}' for target
group '{target_group['TargetGroupName']}'."
            )

            if error_code == "ListenerNotFoundException":
                log.error(
                    f"The listener could not be found for the load balancer
'{load_balancer_name}'. "
                    "Please check the load balancer name and target group
configuration."
                )
            if error_code == "InvalidConfigurationRequestException":
                log.error(
                    f"The configuration provided for the listener on load
balancer '{load_balancer_name}' is invalid. "
```

```
                    "Please review the provided protocol, port, and target group
settings."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_load_balancer(self, load_balancer_name) -> None:
        """
        Deletes a load balancer.

        :param load_balancer_name: The name of the load balancer to delete.
        """
        try:
            response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            lb_arn = response["LoadBalancers"][0]["LoadBalancerArn"]
            self.elb_client.delete_load_balancer(LoadBalancerArn=lb_arn)
            log.info("Deleted load balancer %s.", load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancers_deleted")
            log.info("Waiting for load balancer to be deleted...")
            waiter.wait(Names=[load_balancer_name])
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Couldn't delete load balancer '{load_balancer_name}'. Error
code: {error_code}, Message: {err.response['Error']['Message']}"
            )

            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    f"The load balancer '{load_balancer_name}' does not exist. "
                    "Please check the name and try again."
                )
            log.error(f"Full error:\n\t{err}")


    def get_endpoint(self, load_balancer_name) -> str:
        """
        Gets the HTTP endpoint of the load balancer.

        :return: The endpoint.
        """
        try:
```

```
            response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            return response["LoadBalancers"][0]["DNSName"]
        except ClientError as err:
            log.error(
                f"Couldn't get the endpoint for load balancer
{load_balancer_name}"
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    "Verify load balancer name and ensure it exists in the AWS
console."
                )
            log.error(f"Full error:\n\t{err}")

    @staticmethod
    def verify_load_balancer_endpoint(endpoint) -> bool:
        """
        Verify this computer can successfully send a GET request to the load
balancer endpoint.

        :param endpoint: The endpoint to verify.
        :return: True if the GET request is successful, False otherwise.
        """
        retries = 3
        verified = False
        while not verified and retries > 0:
            try:
                lb_response = requests.get(f"http://{endpoint}")
                log.info(
                    "Got response %s from load balancer endpoint.",
                    lb_response.status_code,
                )
                if lb_response.status_code == 200:
                    verified = True
                else:
                    retries = 0
            except requests.exceptions.ConnectionError:
                log.info(
                    "Got connection error from load balancer endpoint,
retrying..."
                )
```

```
                retries -= 1
                time.sleep(10)
        return verified

    def check_target_health(self, target_group_name: str) -> List[Dict[str,
Any]]:
        """
        Checks the health of the instances in the target group.

        :return: The health status of the target group.
        """
        try:
            tg_response = self.elb_client.describe_target_groups(
                Names=[target_group_name]
            )
            health_response = self.elb_client.describe_target_health(
                TargetGroupArn=tg_response["TargetGroups"][0]["TargetGroupArn"]
            )
        except ClientError as err:
            log.error(f"Couldn't check health of {target_group_name} target(s).")
            error_code = err.response["Error"]["Code"]
            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    "Load balancer associated with the target group was not
found. "
                    "Ensure the load balancer exists, is in the correct AWS
region, and "
                    "that you have the necessary permissions to access it.",
                )
            elif error_code == "TargetGroupNotFoundException":
                log.error(
                    "Target group was not found. "
                    "Verify the target group name, check that it exists in the
correct region, "
                    "and ensure it has not been deleted or created in a different
account.",
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return health_response["TargetHealthDescriptions"]
```

Créez une classe qui utilise DynamoDB pour simuler un service de recommandation.

```
class RecommendationService:
    """
    Encapsulates a DynamoDB table to use as a service that recommends books,
 movies,
    and songs.
    """

    def __init__(self, table_name: str, dynamodb_client: boto3.client):
        """
        Initializes the RecommendationService class with the necessary
 parameters.

        :param table_name: The name of the DynamoDB recommendations table.
        :param dynamodb_client: A Boto3 DynamoDB client.
        """
        self.table_name = table_name
        self.dynamodb_client = dynamodb_client

    def create(self) -> Dict[str, Any]:
        """
        Creates a DynamoDB table to use as a recommendation service. The table
 has a
        hash key named 'MediaType' that defines the type of media recommended,
 such as
        Book or Movie, and a range key named 'ItemId' that, combined with the
MediaType,
        forms a unique identifier for the recommended item.

        :return: Data about the newly created table.
        :raises RecommendationServiceError: If the table creation fails.
        """
        try:
            response = self.dynamodb_client.create_table(
                TableName=self.table_name,
                AttributeDefinitions=[
                    {"AttributeName": "MediaType", "AttributeType": "S"},
                    {"AttributeName": "ItemId", "AttributeType": "N"},
                ],
                KeySchema=[
```

```
                    {"AttributeName": "MediaType", "KeyType": "HASH"},
                    {"AttributeName": "ItemId", "KeyType": "RANGE"},
                ],
                ProvisionedThroughput={"ReadCapacityUnits": 5,
"WriteCapacityUnits": 5},
            )
            log.info("Creating table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s created.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceInUseException":
                log.info("Table %s exists, nothing to be done.", self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when creating table: {err}."
                )
        else:
            return response

    def populate(self, data_file: str) -> None:
        """
        Populates the recommendations table from a JSON file.

        :param data_file: The path to the data file.
        :raises RecommendationServiceError: If the table population fails.
        """
        try:
            with open(data_file) as data:
                items = json.load(data)
            batch = [{"PutRequest": {"Item": item}} for item in items]
            self.dynamodb_client.batch_write_item(RequestItems={self.table_name:
batch})
            log.info(
                "Populated table %s with items from %s.", self.table_name,
data_file
            )
        except ClientError as err:
            raise RecommendationServiceError(
                self.table_name, f"Couldn't populate table from {data_file}:
{err}"
            )

    def destroy(self) -> None:
```

```
        """
        Deletes the recommendations table.

        :raises RecommendationServiceError: If the table deletion fails.
        """
        try:
            self.dynamodb_client.delete_table(TableName=self.table_name)
            log.info("Deleting table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_not_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s deleted.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                log.info("Table %s does not exist, nothing to do.",
 self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when deleting table: {err}."
                )
```

Créez une classe qui englobe les actions de Systems Manager.

```
class ParameterHelper:
    """
    Encapsulates Systems Manager parameters. This example uses these parameters
 to drive
    the demonstration of resilient architecture, such as failure of a dependency
 or
    how the service responds to a health check.
    """

    table: str = "doc-example-resilient-architecture-table"
    failure_response: str = "doc-example-resilient-architecture-failure-response"
    health_check: str = "doc-example-resilient-architecture-health-check"

    def __init__(self, table_name: str, ssm_client: boto3.client):
        """
        Initializes the ParameterHelper class with the necessary parameters.
```

```
        :param table_name: The name of the DynamoDB table that is used as a
recommendation
                           service.
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.table_name = table_name

    def reset(self) -> None:
        """
        Resets the Systems Manager parameters to starting values for the demo.
        These are the name of the DynamoDB recommendation table, no response when
a
        dependency fails, and shallow health checks.
        """
        self.put(self.table, self.table_name)
        self.put(self.failure_response, "none")
        self.put(self.health_check, "shallow")

    def put(self, name: str, value: str) -> None:
        """
        Sets the value of a named Systems Manager parameter.

        :param name: The name of the parameter.
        :param value: The new value of the parameter.
        :raises ParameterHelperError: If the parameter value cannot be set.
        """
        try:
            self.ssm_client.put_parameter(
                Name=name, Value=value, Overwrite=True, Type="String"
            )
            log.info("Setting parameter %s to '%s'.", name, value)
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to set parameter {name}.")
            if error_code == "ParameterLimitExceeded":
                log.error(
                    "The parameter limit has been exceeded. "
                    "Consider deleting unused parameters or request a limit
increase."
                )
            elif error_code == "ParameterAlreadyExists":
                log.error(
```

```
                    "The parameter already exists and overwrite is set to False.
"
                    "Use Overwrite=True to update the parameter."
                )
            log.error(f"Full error:\n\t{err}")
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans AWS SDK for Python (Boto3) API Reference.
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadBalancers
  - DescribeSubnets
  - DescribeTargetGroups
  - DescribeTargetHealth
  - DescribeVpcs
  - RebootInstances

- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Configuration de la connectivité des services de conteneurs

L'exemple de code suivant illustre comment :

- Création de l'infrastructure VPC
- Configuration de la journalisation
- Création du cluster ECS
- Configuration des rôles IAM
- Créez le service avec Service Connect
- Vérification du déploiement
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# ECS Service Connect Tutorial Script v4 - Modified to use Default VPC
# This script creates an ECS cluster with Service Connect and deploys an nginx
 service
# Uses the default VPC to avoid VPC limits
```

```
set -e  # Exit on any error

# Configuration
SCRIPT_NAME="ECS Service Connect Tutorial"
LOG_FILE="ecs-service-connect-tutorial-v4-default-vpc.log"
REGION=${AWS_DEFAULT_REGION:-${AWS_REGION:-$(aws configure get region 2>/dev/
null)}}
if [ -z "$REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Set one with: aws configure set region us-east-1"
    exit 1
fi
ENV_PREFIX="tutorial"
CLUSTER_NAME="${ENV_PREFIX}-cluster"
NAMESPACE_NAME="service-connect"

# Generate random suffix for unique resource names
RANDOM_SUFFIX=$(openssl rand -hex 6)

# Arrays to track created resources for cleanup
declare -a CREATED_RESOURCES=()

# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
}

# Error handling function
handle_error() {
    log "ERROR: Script failed at line $1"
    log "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set up error handling
trap 'handle_error $LINENO' ERR

# Function to add resource to tracking array
track_resource() {
    CREATED_RESOURCES+=("$1")
    log "Tracking resource: $1"
}
```

```
# Function to check if command output contains actual errors
check_for_errors() {
    local output="$1"
    local command_name="$2"

    # Check for specific AWS CLI error patterns, not just any occurrence of
 "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameterException\|
AccessDenied\|ValidationException\|ResourceNotFoundException"; then
        log "ERROR in $command_name: $output"
        return 1
    fi
    return 0
}

# Function to get AWS account ID
get_account_id() {
    ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
    log "Using AWS Account ID: $ACCOUNT_ID"
}

# Function to wait for resources to be ready
wait_for_resource() {
    local resource_type="$1"
    local resource_id="$2"

    case "$resource_type" in
        "cluster")
            log "Waiting for cluster $resource_id to be active..."
            local attempt=1
            local max_attempts=30
            while [ $attempt -le $max_attempts ]; do
                local status=$(aws ecs describe-clusters --clusters
 "$resource_id" --query 'clusters[0].status' --output text)
                if [ "$status" = "ACTIVE" ]; then
                    log "Cluster is now active"
                    return 0
                fi
                log "Cluster status: $status (attempt $attempt/$max_attempts)"
                sleep 10
                ((attempt++))
            done
            log "ERROR: Cluster did not become active within expected time"
            return 1
```

```
            ;;
        "service")
            log "Waiting for service $resource_id to be stable..."
            aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id"
            ;;
        "nat-gateway")
            log "Waiting for NAT Gateway $resource_id to be available..."
            aws ec2 wait nat-gateway-available --nat-gateway-ids "$resource_id"
            ;;
    esac
}

# Function to use default VPC infrastructure
setup_default_vpc_infrastructure() {
    log "Using default VPC infrastructure..."

    # Get default VPC
    VPC_ID=$(aws ec2 describe-vpcs --filters "Name=isDefault,Values=true" --query
 'Vpcs[0].VpcId' --output text)
    if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
        log "ERROR: No default VPC found. Please create a default VPC first."
        exit 1
    fi
    log "Using default VPC: $VPC_ID"

    # Get default subnets
    SUBNETS=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 "Name=default-for-az,Values=true" --query 'Subnets[].SubnetId' --output text)
    SUBNET_ARRAY=($SUBNETS)

    if [ ${#SUBNET_ARRAY[@]} -lt 2 ]; then
        log "ERROR: Need at least 2 subnets for ECS Service Connect. Found:
 ${#SUBNET_ARRAY[@]}"
        exit 1
    fi

    PUBLIC_SUBNET1=${SUBNET_ARRAY[0]}
    PUBLIC_SUBNET2=${SUBNET_ARRAY[1]}

    log "Using subnets: $PUBLIC_SUBNET1, $PUBLIC_SUBNET2"

    # Create security group for ECS tasks
    SG_OUTPUT=$(aws ec2 create-security-group \
```

```
        --group-name "${ENV_PREFIX}-ecs-sg-${RANDOM_SUFFIX}" \
        --description "Security group for ECS Service Connect tutorial" \
        --vpc-id "$VPC_ID" \
        --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-ecs-service-
connect}]' 2>&1)
    check_for_errors "$SG_OUTPUT" "create-security-group"
    SECURITY_GROUP_ID=$(echo "$SG_OUTPUT" | grep -o '"GroupId": "[^"]*"' | cut -
d'"' -f4)
    track_resource "SG:$SECURITY_GROUP_ID"
    log "Created security group: $SECURITY_GROUP_ID"

    # Add inbound rules to security group
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 443 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    log "Default VPC infrastructure setup completed"
}

# Function to create CloudWatch log groups
create_log_groups() {
    log "Creating CloudWatch log groups..."

    # Create log group for nginx container
    aws logs create-log-group --log-group-name "/ecs/service-connect-nginx"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-nginx created"
            track_resource "LOG_GROUP:/ecs/service-connect-nginx"
        else
            log "Log group /ecs/service-connect-nginx already exists"
        fi
    }
```

```
    # Create log group for service connect proxy
    aws logs create-log-group --log-group-name "/ecs/service-connect-proxy"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-proxy created"
            track_resource "LOG_GROUP:/ecs/service-connect-proxy"
        else
            log "Log group /ecs/service-connect-proxy already exists"
        fi
    }
}

# Function to create ECS cluster with Service Connect
create_ecs_cluster() {
    log "Creating ECS cluster with Service Connect..."

    CLUSTER_OUTPUT=$(aws ecs create-cluster \
        --cluster-name "$CLUSTER_NAME" \
        --service-connect-defaults namespace="$NAMESPACE_NAME" \
        --tags key=Environment,value=tutorial key=project,value=doc-smith
 key=tutorial,value=amazon-ecs-service-connect 2>&1)
    check_for_errors "$CLUSTER_OUTPUT" "create-cluster"

    track_resource "CLUSTER:$CLUSTER_NAME"
    log "Created ECS cluster: $CLUSTER_NAME"

    wait_for_resource "cluster" "$CLUSTER_NAME"

    # Track the Service Connect namespace that gets created
    # Wait a moment for the namespace to be created
    sleep 5
    NAMESPACE_ID=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_ID" && "$NAMESPACE_ID" != "None" ]]; then
        track_resource "NAMESPACE:$NAMESPACE_ID"
        log "Service Connect namespace created: $NAMESPACE_ID"
    fi
}

# Function to create IAM roles
```

```
create_iam_roles() {
    log "Creating IAM roles..."

    # Check if ecsTaskExecutionRole exists
    if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
        log "IAM role ecsTaskExecutionRole exists"
    else
        log "Creating ecsTaskExecutionRole..."
        aws iam create-role \
            --role-name ecsTaskExecutionRole \
            --assume-role-policy-document '{
                "Version":"2012-10-17",
                "Statement": [{
                    "Effect": "Allow",
                    "Principal": {"Service": "ecs-tasks.amazonaws.com"},
                    "Action": "sts:AssumeRole"
                }]
            }' >/dev/null 2>&1
        aws iam attach-role-policy \
            --role-name ecsTaskExecutionRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy >/dev/null 2>&1
        track_resource "ROLE:ecsTaskExecutionRole"
        aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-service-connect
        log "Created ecsTaskExecutionRole"
        sleep 10
    fi

    # Check if ecsTaskRole exists, create if not
    if aws iam get-role --role-name ecsTaskRole >/dev/null 2>&1; then
        log "IAM role ecsTaskRole exists"
    else
        log "IAM role ecsTaskRole does not exist, will create it"

        # Create trust policy for ECS tasks
        cat > /tmp/ecs-task-trust-policy.json << EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
```

```
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

        aws iam create-role \
            --role-name ecsTaskRole \
            --assume-role-policy-document file:///tmp/ecs-task-trust-policy.json
 >/dev/null

        track_resource "IAM_ROLE:ecsTaskRole"
        aws iam tag-role --role-name ecsTaskRole --tags Key=project,Value=doc-
smith Key=tutorial,Value=amazon-ecs-service-connect
        log "Created ecsTaskRole"

        # Wait for role to be available
        sleep 10
    fi
}

# Function to create task definition
create_task_definition() {
    log "Creating task definition..."

    # Create task definition JSON
    cat > /tmp/task-definition.json << EOF
{
    "family": "service-connect-nginx",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole",
    "taskRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskRole",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "portMappings": [
                {
                    "containerPort": 80,
                    "protocol": "tcp",
```

```
                    "name": "nginx-port"
                }
            ],
            "essential": true,
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/service-connect-nginx",
                    "awslogs-region": "${REGION}",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

    TASK_DEF_OUTPUT=$(aws ecs register-task-definition --cli-input-json file:///
tmp/task-definition.json 2>&1)
    check_for_errors "$TASK_DEF_OUTPUT" "register-task-definition"

    TASK_DEF_ARN=$(echo "$TASK_DEF_OUTPUT" | grep -o '"taskDefinitionArn":
 "[^"]*"' | cut -d'"' -f4)
    track_resource "TASK_DEF:service-connect-nginx"
    log "Created task definition: $TASK_DEF_ARN"

    # Clean up temporary file
    rm -f /tmp/task-definition.json
}

# Function to create ECS service with Service Connect
create_ecs_service() {
    log "Creating ECS service with Service Connect..."

    # Create service definition JSON
    cat > /tmp/service-definition.json << EOF
{
    "serviceName": "service-connect-nginx-service",
    "cluster": "${CLUSTER_NAME}",
    "taskDefinition": "service-connect-nginx",
    "desiredCount": 1,
    "launchType": "FARGATE",
    "networkConfiguration": {
        "awsvpcConfiguration": {
```

```
            "subnets": ["${PUBLIC_SUBNET1}", "${PUBLIC_SUBNET2}"],
            "securityGroups": ["${SECURITY_GROUP_ID}"],
            "assignPublicIp": "ENABLED"
        }
    },
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "${NAMESPACE_NAME}",
        "services": [
            {
                "portName": "nginx-port",
                "discoveryName": "nginx",
                "clientAliases": [
                    {
                        "port": 80,
                        "dnsName": "nginx"
                    }
                ]
            }
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-proxy",
                "awslogs-region": "${REGION}",
                "awslogs-stream-prefix": "ecs-service-connect"
            }
        }
    },
    "tags": [
        {
            "key": "Environment",
            "value": "tutorial"
        },
        {
            "key": "project",
            "value": "doc-smith"
        },
        {
            "key": "tutorial",
            "value": "amazon-ecs-service-connect"
        }
    ]
}
```

```
EOF

    SERVICE_OUTPUT=$(aws ecs create-service --cli-input-json file:///tmp/service-
definition.json 2>&1)
    check_for_errors "$SERVICE_OUTPUT" "create-service"

    track_resource "SERVICE:service-connect-nginx-service"
    log "Created ECS service: service-connect-nginx-service"

    wait_for_resource "service" "service-connect-nginx-service"

    # Clean up temporary file
    rm -f /tmp/service-definition.json
}

# Function to verify deployment
verify_deployment() {
    log "Verifying deployment..."

    # Check service status
    SERVICE_STATUS=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].status' --output text)
    log "Service status: $SERVICE_STATUS"

    # Check running tasks
    RUNNING_COUNT=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].runningCount' --output text)
    log "Running tasks: $RUNNING_COUNT"

    # Get task ARN
    TASK_ARN=$(aws ecs list-tasks \
        --cluster "$CLUSTER_NAME" \
        --service-name "service-connect-nginx-service" \
        --query 'taskArns[0]' --output text)

    if [[ "$TASK_ARN" != "None" && -n "$TASK_ARN" ]]; then
        log "Task ARN: $TASK_ARN"

        # Try to get task IP address
        TASK_IP=$(aws ecs describe-tasks \
```

```
            --cluster "$CLUSTER_NAME" \
            --tasks "$TASK_ARN" \
            --query 'tasks[0].attachments[0].details[?
name==`privateIPv4Address`].value' \
            --output text 2>/dev/null || echo "")

        if [[ -n "$TASK_IP" && "$TASK_IP" != "None" ]]; then
            log "Task IP address: $TASK_IP"
        else
            log "Could not retrieve task IP address"
        fi
    fi

    # Check Service Connect namespace
    NAMESPACE_STATUS=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_STATUS" && "$NAMESPACE_STATUS" != "None" ]]; then
        log "Service Connect namespace '$NAMESPACE_NAME' is active"
    else
        log "Service Connect namespace '$NAMESPACE_NAME' not found or not active"
    fi

    # Display Service Connect configuration
    log "Service Connect configuration:"
    aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].serviceConnectConfiguration' 2>/dev/null || true
}

# Function to display created resources
display_resources() {
    echo ""
    echo "============================================"
    echo "CREATED RESOURCES"
    echo "============================================"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "- $resource"
    done
    echo "============================================"
    echo ""
```

```
}

# Function to cleanup resources
cleanup_resources() {
    log "Starting cleanup process..."

    # Delete resources in reverse order of creation
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[i]}"
        resource_type=$(echo "$resource" | cut -d':' -f1)
        resource_id=$(echo "$resource" | cut -d':' -f2)

        log "Cleaning up $resource_type: $resource_id"

        case "$resource_type" in
            "SERVICE")
                aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --desired-count 0 2>&1 | grep -qi "error" && log "Warning: Failed
 to scale down service $resource_id"
                aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id" 2>/dev/null || true
                aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --force 2>&1 | grep -qi "error" && log "Warning: Failed to delete
 service $resource_id"
                ;;
            "TASK_DEF")
                TASK_DEF_ARNS=$(aws ecs list-task-definitions --family-prefix
 "$resource_id" --query 'taskDefinitionArns' --output text 2>/dev/null)
                for arn in $TASK_DEF_ARNS; do
                    aws ecs deregister-task-definition --task-definition "$arn"
 >/dev/null 2>&1 || true
                done
                ;;
            "ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "IAM_ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
```

```
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "CLUSTER")
                aws ecs delete-cluster --cluster "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete cluster $resource_id"
                ;;
            "SG")
                for attempt in 1 2 3 4 5; do
                    if aws ec2 delete-security-group --group-id "$resource_id"
 2>/dev/null; then
                        break
                    fi
                    log "Security group $resource_id still has dependencies,
 retrying in 30s ($attempt/5)..."
                    sleep 30
                done
                ;;
            "LOG_GROUP")
                aws logs delete-log-group --log-group-name "$resource_id" 2>&1 |
 grep -qi "error" && log "Warning: Failed to delete log group $resource_id"
                ;;
            "NAMESPACE")
                # First, delete any services in the namespace
                NAMESPACE_SERVICES=$(aws servicediscovery list-services \
                    --filters Name=NAMESPACE_ID,Values="$resource_id" \
                    --query 'Services[].Id' --output text 2>/dev/null || echo "")

                if [[ -n "$NAMESPACE_SERVICES" && "$NAMESPACE_SERVICES" !=
 "None" ]]; then
                    for service_id in $NAMESPACE_SERVICES; do
                        aws servicediscovery delete-service --id "$service_id" >/
dev/null 2>&1 || true
                        sleep 2
                    done
                fi

                # Then delete the namespace
                aws servicediscovery delete-namespace --id "$resource_id" >/dev/
null 2>&1 || true
                ;;
        esac

        sleep 2  # Brief pause between deletions
```

```
    done

    # Clean up temporary files
    rm -f /tmp/ecs-task-trust-policy.json
    rm -f /tmp/task-definition.json
    rm -f /tmp/service-definition.json

    log "Cleanup completed"
}

# Main execution
main() {
    log "Starting $SCRIPT_NAME v4 (Default VPC)"
    log "Region: $REGION"
    log "Log file: $LOG_FILE"

    # Get AWS account ID
    get_account_id

    # Setup infrastructure using default VPC
    setup_default_vpc_infrastructure

    # Create CloudWatch log groups
    create_log_groups

    # Create ECS cluster
    create_ecs_cluster

    # Create IAM roles
    create_iam_roles

    # Create task definition
    create_task_definition

    # Create ECS service
    create_ecs_service

    # Verify deployment
    verify_deployment

    log "Tutorial completed successfully!"

    # Display created resources
    display_resources
```

```
    # Ask user if they want to clean up
    echo ""
    echo "==========================================="
    echo "CLEANUP CONFIRMATION"
    echo "==========================================="
    echo "Do you want to clean up all created resources? (y/n): "
    CLEANUP_CHOICE="y"

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        cleanup_resources
        log "All resources have been cleaned up"
    else
        log "Resources left intact. You can clean them up later by running the
 cleanup function."
        echo ""
        echo "To clean up resources later, you can use the AWS CLI commands or
 the AWS Management Console."
        echo "Remember to delete resources in the correct order to avoid
 dependency issues."
    fi
}

# Make script executable and run
chmod +x "$0"
main "$@"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateLogGroup
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteLogGroup

- DeleteNamespace
- DeleteRole
- DeleteSecurityGroup
- DeleteService (AWS Cloud Map)
- DeleteService (Amazon ECS)
- DeregisterTaskDefinition
- DescribeClusters
- DescribeServices
- DescribeSubnets
- DescribeTasks
- DescribeVpcs
- DetachRolePolicy
- GetCallerIdentity
- GetRole
- ListNamespaces
- ListServices
- ListTaskDefinitions
- ListTasks
- RegisterTaskDefinition
- UpdateService
- Attendre (Amazon EC2)
- Attendre (Amazon ECS)

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Création d'une tâche de conteneur pour le type de lancement sans serveur

L'exemple de code suivant illustre comment :

- Créer le cluster
- Créer une définition de tâche

- Créez le service
- Nettoyage

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Amazon ECS Fargate Tutorial Script - Version 5
# This script creates an ECS cluster, task definition, and service using Fargate
 launch type
# Fixed version with proper resource dependency handling during cleanup
# Security improvements applied

set -e  # Exit on any error

# Initialize logging
LOG_FILE="ecs-fargate-tutorial-v5.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon ECS Fargate tutorial at $(date)"
echo "Log file: $LOG_FILE"

# Generate random identifier for unique resource names
RANDOM_ID=$(openssl rand -hex 6)
CLUSTER_NAME="fargate-cluster-${RANDOM_ID}"
SERVICE_NAME="fargate-service-${RANDOM_ID}"
TASK_FAMILY="sample-fargate-${RANDOM_ID}"
SECURITY_GROUP_NAME="ecs-fargate-sg-${RANDOM_ID}"

# Array to track created resources for cleanup
CREATED_RESOURCES=()

# Function to log and execute commands with input validation
```

```
execute_command() {
    local cmd="$1"
    local description="$2"

    # Validate that cmd is not empty
    if [[ -z "$cmd" ]]; then
        echo "ERROR: Command is empty"
        return 1
    fi

    echo ""
    echo "============================================"
    echo "EXECUTING: $description"
    echo "COMMAND: $cmd"
    echo "============================================"

    local output
    local exit_code
    set +e  # Temporarily disable exit on error
    output=$(eval "$cmd" 2>&1)
    exit_code=$?
    set -e  # Re-enable exit on error

    if [[ $exit_code -eq 0 ]]; then
        echo "SUCCESS: $description"
        echo "OUTPUT: $output"
        return 0
    else
        echo "FAILED: $description"
        echo "EXIT CODE: $exit_code"
        echo "OUTPUT: $output"
        return 1
    fi
}

# Function to check for actual AWS API errors in command output
check_for_aws_errors() {
    local output="$1"
    local description="$2"

    # Look for specific AWS error patterns, not just the word "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameter\|
AccessDenied\|ResourceNotFound\|ValidationException"; then
        echo "AWS API ERROR detected in output for: $description"
```

```
        echo "Output: $output"
        return 1
    fi
    return 0
}

# Function to safely extract JSON values
safe_json_extract() {
    local json="$1"
    local key="$2"
    local value

    value=$(echo "$json" | grep -o "\"$key\": \"[^\"]*\"" | cut -d'"' -f4 2>/dev/
null || echo "")
    echo "$value"
}

# Function to wait for network interfaces to be cleaned up
wait_for_network_interfaces_cleanup() {
    local security_group_id="$1"
    local max_attempts=30
    local attempt=1

    # Validate security group ID format
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
        echo "ERROR: Invalid security group ID format: $security_group_id"
        return 1
    fi

    echo "Waiting for network interfaces to be cleaned up..."

    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Checking for dependent network
 interfaces..."

        # Check if there are any network interfaces still using this security
 group
        local eni_count
        eni_count=$(aws ec2 describe-network-interfaces \
            --filters "Name=group-id,Values=$security_group_id" \
            --query "length(NetworkInterfaces)" \
            --output text 2>/dev/null || echo "0")

        if [[ "$eni_count" == "0" ]]; then
```

```
            echo "No network interfaces found using security group
 $security_group_id"
            return 0
        else
            echo "Found $eni_count network interface(s) still using security
 group $security_group_id"
            echo "Waiting 10 seconds before next check..."
            sleep 10
            ((attempt++))
        fi
    done

    echo "WARNING: Network interfaces may still be attached after $max_attempts
 attempts"
    echo "This is normal and the security group deletion will be retried"
    return 1
}

# Function to retry security group deletion with exponential backoff
retry_security_group_deletion() {
    local security_group_id="$1"
    local max_attempts=10
    local attempt=1
    local wait_time=5

    # Validate security group ID format
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
        echo "ERROR: Invalid security group ID format: $security_group_id"
        return 1
    fi

    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Trying to delete security group
 $security_group_id"

        if execute_command "aws ec2 delete-security-group --group-id
 '$security_group_id'" "Delete security group (attempt $attempt)"; then
            echo "Successfully deleted security group $security_group_id"
            return 0
        else
            if [[ $attempt -eq $max_attempts ]]; then
                echo "FAILED: Could not delete security group $security_group_id
 after $max_attempts attempts"
```

```
                echo "This may be due to network interfaces that are still being
 cleaned up by AWS"
                echo "You can manually delete it later using: aws ec2 delete-
security-group --group-id $security_group_id"
                return 1
            else
                echo "Waiting $wait_time seconds before retry..."
                sleep $wait_time
                wait_time=$((wait_time * 2))  # Exponential backoff
                ((attempt++))
            fi
        fi
    done
}

# Function to cleanup resources with proper dependency handling
cleanup_resources() {
    echo ""
    echo "==========================================="
    echo "CLEANUP PROCESS"
    echo "==========================================="
    echo "The following resources were created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
    echo ""
    echo "Auto-confirming cleanup of all created resources..."

    CLEANUP_CHOICE="y"

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        echo "Starting cleanup process..."

        # Step 1: Scale service to 0 tasks first, then delete service
        if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Service: $SERVICE_NAME " ]];
 then
            echo ""
            echo "Step 1: Scaling service to 0 tasks..."
            if execute_command "aws ecs update-service --cluster '$CLUSTER_NAME'
 --service '$SERVICE_NAME' --desired-count 0" "Scale service to 0 tasks"; then
                echo "Waiting for service to stabilize after scaling to 0..."
                execute_command "aws ecs wait services-stable --cluster
 '$CLUSTER_NAME' --services '$SERVICE_NAME'" "Wait for service to stabilize"
```

```
                echo "Deleting service..."
                execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME'" "Delete ECS service"
            else
                echo "WARNING: Failed to scale service. Attempting to delete
anyway..."
                execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME' --force" "Force delete ECS service"
            fi
        fi

        # Step 2: Wait a bit for tasks to fully terminate
        echo ""
        echo "Step 2: Waiting for tasks to fully terminate..."
        sleep 15

        # Step 3: Delete cluster
        if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Cluster: $CLUSTER_NAME " ]];
then
            echo ""
            echo "Step 3: Deleting cluster..."
            execute_command "aws ecs delete-cluster --cluster '$CLUSTER_NAME'"
"Delete ECS cluster"
        fi

        # Step 4: Wait for network interfaces to be cleaned up, then delete
security group
        if [[ -n "$SECURITY_GROUP_ID" && "$SECURITY_GROUP_ID" != "None" ]]; then
            echo ""
            echo "Step 4: Cleaning up security group..."

            # First, wait for network interfaces to be cleaned up
            wait_for_network_interfaces_cleanup "$SECURITY_GROUP_ID"

            # Then retry security group deletion with backoff
            retry_security_group_deletion "$SECURITY_GROUP_ID"
        fi

        # Step 5: Clean up task definition (deregister all revisions)
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
" ]]; then
            echo ""
            echo "Step 5: Deregistering task definition revisions..."
```

```
            # Get all revisions of the task definition
            local revisions
            revisions=$(aws ecs list-task-definitions --family-prefix
 "$TASK_FAMILY" --query "taskDefinitionArns" --output text 2>/dev/null || echo
 "")

            if [[ -n "$revisions" && "$revisions" != "None" ]]; then
                for revision_arn in $revisions; do
                    echo "Deregistering task definition: $revision_arn"
                    execute_command "aws ecs deregister-task-definition --task-
definition '$revision_arn'" "Deregister task definition $revision_arn" || true
                done
            else
                echo "No task definition revisions found to deregister"
            fi
        fi

        echo ""
        echo "============================================"
        echo "CLEANUP COMPLETED"
        echo "============================================"
        echo "All resources have been cleaned up successfully!"

    else
        echo "Cleanup skipped. Resources remain active."
        echo ""
        echo "To clean up manually later, use the following commands in order:"
        echo "1. Scale service to 0: aws ecs update-service --cluster
 $CLUSTER_NAME --service $SERVICE_NAME --desired-count 0"
        echo "2. Wait for stability: aws ecs wait services-stable --cluster
 $CLUSTER_NAME --services $SERVICE_NAME"
        echo "3. Delete service: aws ecs delete-service --cluster $CLUSTER_NAME
 --service $SERVICE_NAME"
        echo "4. Delete cluster: aws ecs delete-cluster --cluster $CLUSTER_NAME"
        echo "5. Wait 2-3 minutes, then delete security group: aws ec2 delete-
security-group --group-id $SECURITY_GROUP_ID"
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
 " ]]; then
            echo "6. Deregister task definitions: aws ecs list-task-definitions
 --family-prefix $TASK_FAMILY"
            echo "   Then for each ARN: aws ecs deregister-task-definition --
task-definition <ARN>"
        fi
    fi
```

```
}

# Trap to handle script interruption
trap cleanup_resources EXIT

echo "Using random identifier: $RANDOM_ID"
echo "Cluster name: $CLUSTER_NAME"
echo "Service name: $SERVICE_NAME"
echo "Task family: $TASK_FAMILY"

# Step 1: Ensure ECS task execution role exists
echo ""
echo "============================================="
echo "STEP 1: VERIFY ECS TASK EXECUTION ROLE"
echo "============================================="

ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ ! "$ACCOUNT_ID" =~ ^[0-9]{12}$ ]]; then
    echo "ERROR: Invalid AWS Account ID retrieved: $ACCOUNT_ID"
    exit 1
fi

EXECUTION_ROLE_ARN="arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole"

# Check if role exists
if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
    echo "ECS task execution role already exists"
else
    echo "Creating ECS task execution role..."

    # Create trust policy with strict validation
    cat > trust-policy.json << 'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

```
EOF

    # Validate JSON before using
    if ! jq empty trust-policy.json 2>/dev/null; then
        echo "ERROR: Invalid JSON in trust policy"
        rm -f trust-policy.json
        exit 1
    fi

    execute_command "aws iam create-role --role-name ecsTaskExecutionRole --
assume-role-policy-document file://trust-policy.json" "Create ECS task execution
 role"

    aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-fargate-linux

    execute_command "aws iam attach-role-policy --role-name
 ecsTaskExecutionRole --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy" "Attach ECS task execution policy"

    # Clean up temporary file securely
    shred -vfz -n 3 trust-policy.json 2>/dev/null || rm -f trust-policy.json

    CREATED_RESOURCES+=("IAM Role: ecsTaskExecutionRole")
fi

# Step 2: Create ECS cluster
echo ""
echo "============================================="
echo "STEP 2: CREATE ECS CLUSTER"
echo "============================================="

CLUSTER_OUTPUT=$(execute_command "aws ecs create-cluster --cluster-name
 '$CLUSTER_NAME' --tags key=project,value=doc-smith key=tutorial,value=amazon-
ecs-fargate-linux" "Create ECS cluster")
check_for_aws_errors "$CLUSTER_OUTPUT" "Create ECS cluster"

CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")

# Step 3: Create task definition
echo ""
echo "============================================="
echo "STEP 3: CREATE TASK DEFINITION"
echo "============================================="
```

```
# Create task definition JSON with validated inputs
cat > task-definition.json << EOF
{
    "family": "$TASK_FAMILY",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "$EXECUTION_ROLE_ARN",
    "containerDefinitions": [
        {
            "name": "fargate-app",
            "image": "public.ecr.aws/docker/library/httpd:2.4-alpine",
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": ["sh", "-c"],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container
 in Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html
 && httpd-foreground\""
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/fargate-sample",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF
```

```
# Validate JSON before using
if ! jq empty task-definition.json 2>/dev/null; then
    echo "ERROR: Invalid JSON in task definition"
    rm -f task-definition.json
    exit 1
fi

TASK_DEF_OUTPUT=$(execute_command "aws ecs register-task-definition --cli-input-
json file://task-definition.json" "Register task definition")
check_for_aws_errors "$TASK_DEF_OUTPUT" "Register task definition"

# Clean up temporary file securely
shred -vfz -n 3 task-definition.json 2>/dev/null || rm -f task-definition.json

CREATED_RESOURCES+=("Task Definition: $TASK_FAMILY")

# Step 4: Set up networking
echo ""
echo "==========================================="
echo "STEP 4: SET UP NETWORKING"
echo "==========================================="

# Get default VPC ID
VPC_ID=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true" --query
 "Vpcs[0].VpcId" --output text)
if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
    echo "ERROR: No default VPC found. Please create a default VPC or specify a
 custom VPC."
    exit 1
fi

# Validate VPC ID format
if [[ ! "$VPC_ID" =~ ^vpc-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid VPC ID format: $VPC_ID"
    exit 1
fi

echo "Using default VPC: $VPC_ID"

# Create security group with restricted access
# Note: This allows HTTP access from anywhere for demo purposes
# In production, restrict source to specific IP ranges or security groups
SECURITY_GROUP_OUTPUT=$(execute_command "aws ec2 create-security-group --
group-name '$SECURITY_GROUP_NAME' --description 'Security group for ECS
```

```
 Fargate tutorial - HTTP access' --vpc-id '$VPC_ID' --tag-specifications
 'ResourceType=security-group,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-ecs-fargate-linux}]'" "Create security group")
check_for_aws_errors "$SECURITY_GROUP_OUTPUT" "Create security group"

SECURITY_GROUP_ID=$(echo "$SECURITY_GROUP_OUTPUT" | grep -o '"GroupId": "[^"]*"'
 | head -1 | cut -d'"' -f4)
if [[ -z "$SECURITY_GROUP_ID" || "$SECURITY_GROUP_ID" == "None" ]]; then
    SECURITY_GROUP_ID=$(aws ec2 describe-security-groups --group-names
 "$SECURITY_GROUP_NAME" --filters "Name=vpc-id,Values=$VPC_ID" --query
 "SecurityGroups[0].GroupId" --output text)
fi

# Validate security group ID format
if [[ ! "$SECURITY_GROUP_ID" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid security group ID format: $SECURITY_GROUP_ID"
    exit 1
fi

echo "Created security group: $SECURITY_GROUP_ID"
CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")

# Add HTTP inbound rule
# WARNING: This allows HTTP access from anywhere (0.0.0.0/0)
# In production environments, restrict this to specific IP ranges
execute_command "aws ec2 authorize-security-group-ingress --group-id
 '$SECURITY_GROUP_ID' --protocol tcp --port 80 --cidr 0.0.0.0/0" "Add HTTP
 inbound rule to security group"

# Get subnet IDs from default VPC
echo "Getting subnet IDs from default VPC..."
SUBNET_IDS_RAW=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 --query "Subnets[*].SubnetId" --output text)
if [[ -z "$SUBNET_IDS_RAW" ]]; then
    echo "ERROR: No subnets found in default VPC"
    exit 1
fi

# Convert to proper comma-separated format, handling both spaces and tabs
SUBNET_IDS_COMMA=$(echo "$SUBNET_IDS_RAW" | tr -s '[:space:]' ',' | sed 's/,$//')
echo "Raw subnet IDs: $SUBNET_IDS_RAW"
echo "Formatted subnet IDs: $SUBNET_IDS_COMMA"

# Validate subnet IDs format
```

```
if [[ ! "$SUBNET_IDS_COMMA" =~ ^subnet-[a-z0-9]+(,subnet-[a-z0-9]+)*$ ]]; then
    echo "ERROR: Invalid subnet ID format: $SUBNET_IDS_COMMA"
    exit 1
fi

# Step 5: Create ECS service
echo ""
echo "=========================================="
echo "STEP 5: CREATE ECS SERVICE"
echo "=========================================="

# Create the service with proper JSON formatting for network configuration
SERVICE_CMD="aws ecs create-service --cluster '$CLUSTER_NAME' --service-name
 '$SERVICE_NAME' --task-definition '$TASK_FAMILY' --desired-count 1 --launch-
type FARGATE --network-configuration '{\"awsvpcConfiguration\":{\"subnets
\":[\"$(echo "$SUBNET_IDS_COMMA" | sed 's/,/","/g')\"],\"securityGroups
\":[\"$SECURITY_GROUP_ID\"],\"assignPublicIp\":\"ENABLED\"}}' --tags
 key=project,value=doc-smith key=tutorial,value=amazon-ecs-fargate-linux"

echo "Service creation command: $SERVICE_CMD"

SERVICE_OUTPUT=$(execute_command "$SERVICE_CMD" "Create ECS service")
check_for_aws_errors "$SERVICE_OUTPUT" "Create ECS service"

CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")

# Step 6: Wait for service to stabilize and get public IP
echo ""
echo "=========================================="
echo "STEP 6: WAIT FOR SERVICE AND GET PUBLIC IP"
echo "=========================================="

echo "Waiting for service to stabilize (this may take a few minutes)..."
execute_command "aws ecs wait services-stable --cluster '$CLUSTER_NAME' --
services '$SERVICE_NAME'" "Wait for service to stabilize"

# Get task ARN
TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query "taskArns[0]" --output text)
if [[ "$TASK_ARN" == "None" || -z "$TASK_ARN" ]]; then
    echo "ERROR: No running tasks found for service"
    exit 1
fi
```

```
echo "Task ARN: $TASK_ARN"

# Get network interface ID
ENI_ID=$(aws ecs describe-tasks --cluster "$CLUSTER_NAME" --tasks "$TASK_ARN"
 --query "tasks[0].attachments[0].details[?name=='networkInterfaceId'].value" --
output text)
if [[ "$ENI_ID" == "None" || -z "$ENI_ID" ]]; then
    echo "ERROR: Could not retrieve network interface ID"
    exit 1
fi

# Validate ENI ID format
if [[ ! "$ENI_ID" =~ ^eni-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid network interface ID format: $ENI_ID"
    exit 1
fi

echo "Network Interface ID: $ENI_ID"

# Get public IP
PUBLIC_IP=$(aws ec2 describe-network-interfaces --network-interface-ids "$ENI_ID"
 --query "NetworkInterfaces[0].Association.PublicIp" --output text)
if [[ "$PUBLIC_IP" == "None" || -z "$PUBLIC_IP" ]]; then
    echo "WARNING: No public IP assigned to the task"
    echo "The task may be in a private subnet or public IP assignment failed"
else
    # Validate IP format
    if [[ ! "$PUBLIC_IP" =~ ^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$ ]];
 then
        echo "ERROR: Invalid IP address format: $PUBLIC_IP"
        PUBLIC_IP=""
    else
        echo ""
        echo "========================================="
        echo "SUCCESS! APPLICATION IS RUNNING"
        echo "========================================="
        echo "Your application is available at: http://$PUBLIC_IP"
        echo "You can test it by opening this URL in your browser"
        echo ""
    fi
fi

# Display service information
echo ""
```

```
echo "============================================="
echo "SERVICE INFORMATION"
echo "============================================="
execute_command "aws ecs describe-services --cluster '$CLUSTER_NAME' --services
 '$SERVICE_NAME'" "Get service details"

echo ""
echo "============================================="
echo "TUTORIAL COMPLETED SUCCESSFULLY"
echo "============================================="
echo "Resources created:"
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "  - $resource"
done

if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
    echo ""
    echo "Application URL: http://$PUBLIC_IP"
fi

echo ""
echo "Script completed at $(date)"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteSecurityGroup
  - DeleteService
  - DeregisterTaskDefinition
  - DescribeNetworkInterfaces
  - DescribeSecurityGroups
  - DescribeServices

- DescribeSubnets
- DescribeTasks
- DescribeVpcs
- GetCallerIdentity
- GetRole
- ListTaskDefinitions
- ListTasks
- RegisterTaskDefinition
- UpdateService
- Attente

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Création d'une API REST avec intégration de proxy de fonctions

L'exemple de code suivant illustre comment :

- Création d'un rôle IAM pour l'exécution de Lambda
- Création et déploiement d'une fonction Lambda
- Création d'une API REST
- Configurer l'intégration du proxy Lambda
- Déploiement et test de l'API
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

set -euo pipefail

# Simple API Gateway Lambda Integration Script
# This script creates a REST API with Lambda proxy integration

# Generate random identifiers
FUNCTION_NAME="GetStartedLambdaProxyIntegration-$(openssl rand -hex 4)"
ROLE_NAME="GetStartedLambdaBasicExecutionRole-$(openssl rand -hex 4)"
API_NAME="LambdaProxyAPI-$(openssl rand -hex 4)"

# Get AWS account info
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
REGION=$(aws configure get region || echo "us-east-1")

# Validate inputs
if [[ -z "$ACCOUNT_ID" ]] || [[ -z "$REGION" ]]; then
    echo "Error: Failed to retrieve AWS account information" >&2
    exit 1
fi

echo "Creating Lambda function code..."

# Create Lambda function code with input validation
cat > lambda_function.py << 'EOF'
import json
import logging

logger = logging.getLogger()
logger.setLevel(logging.INFO)

def lambda_handler(event, context):
    try:
        logger.info("Received event: %s", json.dumps(event))

        greeter = 'World'

        # Safely retrieve greeter from query string parameters
        query_params = event.get('queryStringParameters') or {}
        if isinstance(query_params, dict) and 'greeter' in query_params:
            greeter_value = query_params.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
```

```
                greeter = greeter_value

        # Safely retrieve greeter from multi-value headers
        multi_headers = event.get('multiValueHeaders') or {}
        if isinstance(multi_headers, dict) and 'greeter' in multi_headers:
            greeter_list = multi_headers.get('greeter', [])
            if isinstance(greeter_list, list) and greeter_list:
                greeter = " and ".join(str(g) for g in greeter_list if g)

        # Safely retrieve greeter from headers
        headers = event.get('headers') or {}
        if isinstance(headers, dict) and 'greeter' in headers:
            greeter_value = headers.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
                greeter = greeter_value

        # Safely retrieve greeter from body
        body = event.get('body')
        if body and isinstance(body, str):
            try:
                body_dict = json.loads(body)
                if isinstance(body_dict, dict) and 'greeter' in body_dict:
                    greeter_value = body_dict.get('greeter')
                    if isinstance(greeter_value, str) and greeter_value:
                        greeter = greeter_value
            except (json.JSONDecodeError, ValueError) as e:
                logger.warning("Failed to parse body: %s", str(e))

        # Sanitize greeter to prevent injection
        greeter = greeter.replace('"', '\\"').replace("'", "\\'")

        response = {
            "statusCode": 200,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"message": f"Hello, {greeter}!"})
        }

        logger.info("Response: %s", json.dumps(response))
        return response

    except Exception as e:
        logger.error("Unexpected error: %s", str(e), exc_info=True)
```

```
        return {
            "statusCode": 500,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"error": "Internal server error"})
        }
EOF

# Create deployment package
zip -q function.zip lambda_function.py || {
    echo "Error: Failed to create function.zip" >&2
    exit 1
}

echo "Creating IAM role..."

# Create IAM trust policy
cat > trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role with error handling
aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document file://trust-policy.json \
    --description "Temporary role for Lambda execution" || {
    echo "Error: Failed to create IAM role" >&2
    exit 1
}

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=apigateway-lambda-integration
```

```
# Attach execution policy
aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || {
    echo "Error: Failed to attach IAM policy" >&2
    exit 1
}

# Wait for role propagation
sleep 15

echo "Creating Lambda function..."

# Create Lambda function with Python 3.11 (more recent runtime)
aws lambda create-function \
    --function-name "$FUNCTION_NAME" \
    --runtime python3.11 \
    --role "arn:aws:iam::$ACCOUNT_ID:role/$ROLE_NAME" \
    --handler lambda_function.lambda_handler \
    --zip-file fileb://function.zip \
    --timeout 30 \
    --memory-size 128 \
    --environment "Variables={LOG_LEVEL=INFO}" \
    --tags project=doc-smith,tutorial=apigateway-lambda-integration || {
    echo "Error: Failed to create Lambda function" >&2
    exit 1
}

echo "Creating API Gateway..."

# Create REST API with minimum logging
API_RESPONSE=$(aws apigateway create-rest-api \
    --name "$API_NAME" \
    --endpoint-configuration types=REGIONAL \
    --description "API for Lambda proxy integration tutorial" \
    --tags project=doc-smith,tutorial=apigateway-lambda-integration \
    --output json)

API_ID=$(echo "$API_RESPONSE" | grep -o '"id": "[^"]*"' | head -1 | cut -d'"' -
f4)

if [[ -z "$API_ID" ]]; then
```

```
    echo "Error: Failed to create API Gateway" >&2
    exit 1
fi

# Get root resource ID
ROOT_RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 'items[?path==`/`].id' --output text)

# Create helloworld resource
aws apigateway create-resource \
    --rest-api-id "$API_ID" \
    --parent-id "$ROOT_RESOURCE_ID" \
    --path-part helloworld || {
    echo "Error: Failed to create resource" >&2
    exit 1
}

# Get resource ID
RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 "items[?pathPart=='helloworld'].id" --output text)

# Create ANY method with no authorization (intentional for tutorial)
aws apigateway put-method \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
    --http-method ANY \
    --authorization-type NONE || {
    echo "Error: Failed to create method" >&2
    exit 1
}

# Set up Lambda proxy integration
LAMBDA_URI="arn:aws:apigateway:$REGION:lambda:path/2015-03-31/functions/
arn:aws:lambda:$REGION:$ACCOUNT_ID:function:$FUNCTION_NAME/invocations"

aws apigateway put-integration \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
    --http-method ANY \
    --type AWS_PROXY \
    --integration-http-method POST \
    --uri "$LAMBDA_URI" || {
    echo "Error: Failed to create integration" >&2
    exit 1
```

```
}

# Grant API Gateway permission to invoke Lambda
STATEMENT_ID="apigateway-invoke-$(openssl rand -hex 4)"
SOURCE_ARN="arn:aws:execute-api:$REGION:$ACCOUNT_ID:$API_ID/*/*"

aws lambda add-permission \
    --function-name "$FUNCTION_NAME" \
    --statement-id "$STATEMENT_ID" \
    --action lambda:InvokeFunction \
    --principal apigateway.amazonaws.com \
    --source-arn "$SOURCE_ARN" || {
    echo "Error: Failed to add Lambda permission" >&2
    exit 1
}

# Deploy API
aws apigateway create-deployment \
    --rest-api-id "$API_ID" \
    --stage-name test \
    --description "Test deployment" || {
    echo "Error: Failed to deploy API" >&2
    exit 1
}

echo "Testing API..."

# Test the API
INVOKE_URL="https://$API_ID.execute-api.$REGION.amazonaws.com/test/helloworld"

echo "API URL: $INVOKE_URL"

# Test with query parameter (with proper URL encoding)
echo "Testing with query parameter:"
curl -s -X GET "$INVOKE_URL?greeter=John" | jq . 2>/dev/null || curl -s -X GET
 "$INVOKE_URL?greeter=John"
echo ""

# Test with header
echo "Testing with header:"
curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -H 'greeter: John' | jq . 2>/dev/null || curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
```

```
    -H 'greeter: John'
echo ""

# Test with body
echo "Testing with POST body:"
curl -s -X POST "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}' | jq . 2>/dev/null || curl -s -X POST "$INVOKE_URL"
 \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}'
echo ""

echo "Tutorial completed! API is available at: $INVOKE_URL"

# Cleanup
echo "Cleaning up resources..."

# Delete API
aws apigateway delete-rest-api --rest-api-id "$API_ID" || echo "Warning: Failed
 to delete API" >&2

# Delete Lambda function
aws lambda delete-function --function-name "$FUNCTION_NAME" || echo "Warning:
 Failed to delete Lambda function" >&2

# Detach policy and delete role
aws iam detach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || echo "Warning: Failed to detach policy" >&2

aws iam delete-role --role-name "$ROLE_NAME" || echo "Warning: Failed to delete
 role" >&2

# Clean up local files securely
rm -f lambda_function.py function.zip trust-policy.json

echo "Cleanup completed!"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AddPermission

- AttachRolePolicy
- CreateDeployment
- CreateFunction
- CreateResource
- CreateRestApi
- CreateRole
- DeleteFunction
- DeleteRestApi
- DeleteRole
- DetachRolePolicy
- GetCallerIdentity
- GetResources
- GetRestApis
- PutIntegration
- PutMethod

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Créez des utilisateurs IAM en lecture seule et en lecture-écriture à l'aide d'un AWS Kit SDK

L'exemple de code suivant montre comment créer des utilisateurs et leur associer des politiques.

> **Warning**
>
> Afin d'éviter les risques de sécurité, n'employez pas les utilisateurs IAM pour l'authentification lorsque vous développez des logiciels spécialisés ou lorsque vous travaillez avec des données réelles. Préférez la fédération avec un fournisseur d'identité tel que AWS IAM Identity Center.

- Créez des utilisateurs IAM.

- Attachez une politique pour un utilisateur afin d'obtenir et de placer des objets dans un compartiment Amazon S3.
- Attachez une politique pour le second utilisateur afin d'obtenir des objets depuis le compartiment.
- Obtenez des autorisations différentes du compartiment en fonction des informations d'identification de l'utilisateur.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de l'utilisateur IAM.

```
import logging
import time

import boto3
from botocore.exceptions import ClientError

import access_key_wrapper
import policy_wrapper

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_user(user_name):
    """
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
```

```
        logger.exception("Couldn't create user %s.", user_name)
        raise
    else:
        return user



def update_user(user_name, new_user_name):
    """
    Updates a user's name.

    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user



def list_users():
    """
    Lists the users in the current account.

    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
        logger.exception("Couldn't get users.")
        raise
    else:
        return users



def delete_user(user_name):
```

```
    """
    Deletes a user. Before a user can be deleted, all associated resources,
    such as access keys and policies, must be deleted or detached.

    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
        raise



def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
 user_name)
        raise



def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception(
```

```
            "Couldn't detach policy %s from user %s.", policy_arn, user_name
        )
        raise
```

Créez des fonctions qui encapsulent les actions de politique IAM.

```
import json
import logging
import operator
import pprint
import time

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::amzn-s3-demo-bucket/*' to allow actions on
 all objects
                         in the bucket named 'amzn-s3-demo-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
```

```
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy



def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

Créez des fonctions qui encapsulent les actions de clé d'accès IAM.

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

iam = boto3.resource("iam")

def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
```

```
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair



def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

Utilisez les fonctions d'encapsulation pour créer des utilisateurs avec des politiques différentes et utilisez leurs informations d'identification pour accéder à un compartiment Amazon S3.

```
def usage_demo():
```

```
    """
    Shows how to manage users, keys, and policies.
    This demonstration creates two users: one user who can put and get objects in
an
    Amazon S3 bucket, and another user who can only get objects from the bucket.
    The demo then shows how the users can perform only the actions they are
permitted
    to perform.
    """
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management user demo.")
    print("-" * 88)
    print(
        "Users can have policies and roles attached to grant them specific "
        "permissions."
    )
    s3 = boto3.resource("s3")
    bucket = s3.create_bucket(
        Bucket=f"demo-iam-bucket-{time.time_ns()}",
        CreateBucketConfiguration={
            "LocationConstraint": s3.meta.client.meta.region_name
        },
    )
    print(f"Created an Amazon S3 bucket named {bucket.name}.")
    user_read_writer = create_user("demo-iam-read-writer")
    user_reader = create_user("demo-iam-reader")
    print(f"Created two IAM users: {user_read_writer.name} and
{user_reader.name}")
    update_user(user_read_writer.name, "demo-iam-creator")
    update_user(user_reader.name, "demo-iam-getter")
    users = list_users()
    user_read_writer = next(
        user for user in users if user.user_id == user_read_writer.user_id
    )
    user_reader = next(user for user in users if user.user_id ==
user_reader.user_id)
    print(
        f"Changed the names of the users to {user_read_writer.name} "
        f"and {user_reader.name}."
    )

    read_write_policy = policy_wrapper.create_policy(
        "demo-iam-read-write-policy",
```

```
        "Grants rights to create and get an object in the demo bucket.",
        ["s3:PutObject", "s3:GetObject"],
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(
        f"Created policy {read_write_policy.policy_name} with ARN:
{read_write_policy.arn}"
    )
    print(read_write_policy.description)
    read_policy = policy_wrapper.create_policy(
        "demo-iam-read-policy",
        "Grants rights to get an object from the demo bucket.",
        "s3:GetObject",
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(f"Created policy {read_policy.policy_name} with ARN:
{read_policy.arn}")
    print(read_policy.description)
    attach_policy(user_read_writer.name, read_write_policy.arn)
    print(f"Attached {read_write_policy.policy_name} to
{user_read_writer.name}.")
    attach_policy(user_reader.name, read_policy.arn)
    print(f"Attached {read_policy.policy_name} to {user_reader.name}.")

    user_read_writer_key = access_key_wrapper.create_key(user_read_writer.name)
    print(f"Created access key pair for {user_read_writer.name}.")
    user_reader_key = access_key_wrapper.create_key(user_reader.name)
    print(f"Created access key pair for {user_reader.name}.")

    s3_read_writer_resource = boto3.resource(
        "s3",
        aws_access_key_id=user_read_writer_key.id,
        aws_secret_access_key=user_read_writer_key.secret,
    )
    demo_object_key = f"object-{time.time_ns()}"
    demo_object = None
    while demo_object is None:
        try:
            demo_object = s3_read_writer_resource.Bucket(bucket.name).put_object(
                Key=demo_object_key, Body=b"AWS IAM demo object content!"
            )
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
```

```
                time.sleep(1)
            else:
                raise
    print(
        f"Put {demo_object_key} into {bucket.name} using "
        f"{user_read_writer.name}'s credentials."
    )

    read_writer_object = s3_read_writer_resource.Bucket(bucket.name).Object(
        demo_object_key
    )
    read_writer_content = read_writer_object.get()["Body"].read()
    print(f"Got object {read_writer_object.key} using read-writer user's
credentials.")
    print(f"Object content: {read_writer_content}")

    s3_reader_resource = boto3.resource(
        "s3",
        aws_access_key_id=user_reader_key.id,
        aws_secret_access_key=user_reader_key.secret,
    )
    demo_content = None
    while demo_content is None:
        try:
            demo_object =
s3_reader_resource.Bucket(bucket.name).Object(demo_object_key)
            demo_content = demo_object.get()["Body"].read()
            print(f"Got object {demo_object.key} using reader user's
credentials.")
            print(f"Object content: {demo_content}")
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
                time.sleep(1)
            else:
                raise

    try:
        demo_object.delete()
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("-" * 88)
            print(
```

```
                "Tried to delete the object using the reader user's credentials.
"
                "Got expected AccessDenied error because the reader is not "
                "allowed to delete objects."
            )
            print("-" * 88)

    access_key_wrapper.delete_key(user_reader.name, user_reader_key.id)
    detach_policy(user_reader.name, read_policy.arn)
    policy_wrapper.delete_policy(read_policy.arn)
    delete_user(user_reader.name)
    print(f"Deleted keys, detached and deleted policy, and deleted
{user_reader.name}.")

    access_key_wrapper.delete_key(user_read_writer.name, user_read_writer_key.id)
    detach_policy(user_read_writer.name, read_write_policy.arn)
    policy_wrapper.delete_policy(read_write_policy.arn)
    delete_user(user_read_writer.name)
    print(
        f"Deleted keys, detached and deleted policy, and deleted
{user_read_writer.name}."
    )

    bucket.objects.delete()
    bucket.delete()
    print(f"Emptied and deleted {bucket.name}.")
    print("Thanks for watching!")
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK for Python (Boto3).
  - AttachUserPolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteUser

- DetachUserPolicy
- ListUsers
- UpdateUser

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Création d'un service de conteneur pour les instances de machines virtuelles

L'exemple de code suivant illustre comment :

- Création d'un cluster ECS
- Création et surveillance d'un service
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# ECS EC2 Launch Type Tutorial Script - UPDATED VERSION
# This script demonstrates creating an ECS cluster, launching a container
 instance,
# registering a task definition, and creating a service using the EC2 launch
 type.
# Updated to match the tutorial draft with nginx web server and service creation.
#
# - UPDATED: Changed from sleep task to nginx web server with service

set -e  # Exit on any error
```

```
# Configuration
SCRIPT_NAME="ecs-ec2-tutorial"
LOG_FILE="${SCRIPT_NAME}-$(date +%Y%m%d-%H%M%S).log"
CLUSTER_NAME="tutorial-cluster-$(openssl rand -hex 4)"
TASK_FAMILY="nginx-task-$(openssl rand -hex 4)"
SERVICE_NAME="nginx-service-$(openssl rand -hex 4)"
KEY_PAIR_NAME="ecs-tutorial-key-$(openssl rand -hex 4)"
SECURITY_GROUP_NAME="ecs-tutorial-sg-$(openssl rand -hex 4)"

# Tags
PROJECT_TAG="doc-smith"
TUTORIAL_TAG="ecs-ec2"

# Get current AWS region dynamically
AWS_REGION=$(aws configure get region || echo "us-east-1")

# Resource tracking arrays
CREATED_RESOURCES=()
CLEANUP_ORDER=()

# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
}

# Error handling function
handle_error() {
    local exit_code=$?
    log "ERROR: Script failed with exit code $exit_code"
    log "ERROR: Last command: $BASH_COMMAND"

    echo ""
    echo "========================================="
    echo "ERROR OCCURRED - ATTEMPTING CLEANUP"
    echo "========================================="
    echo "Resources created before error:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done

    cleanup_resources
    exit $exit_code
}
```

```
# Set error trap
trap handle_error ERR

# FIXED: Enhanced cleanup function with proper error handling and logging
cleanup_resources() {
    log "Starting cleanup process..."
    local cleanup_errors=0

    # Delete service first (this will stop tasks automatically)
    if [[ -n "${SERVICE_ARN:-}" ]]; then
        log "Updating service to desired count 0: $SERVICE_NAME"
        if ! aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" --desired-count 0 2>>"$LOG_FILE"; then
            log "WARNING: Failed to update service desired count to 0"
            ((cleanup_errors++))
        else
            log "Waiting for service tasks to stop..."
            sleep 30  # Give time for tasks to stop
        fi

        log "Deleting service: $SERVICE_NAME"
        if ! aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to delete service $SERVICE_NAME"
            ((cleanup_errors++))
        fi
    fi

    # Stop and delete any remaining tasks
    if [[ -n "${TASK_ARN:-}" ]]; then
        log "Stopping task: $TASK_ARN"
        if ! aws ecs stop-task --cluster "$CLUSTER_NAME" --task "$TASK_ARN" --
reason "Tutorial cleanup" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to stop task $TASK_ARN"
            ((cleanup_errors++))
        else
            log "Waiting for task to stop..."
            if ! aws ecs wait tasks-stopped --cluster "$CLUSTER_NAME" --tasks
 "$TASK_ARN" 2>>"$LOG_FILE"; then
                log "WARNING: Task stop wait failed for $TASK_ARN"
                ((cleanup_errors++))
            fi
        fi
    fi
```

```
   # Deregister task definition
   if [[ -n "${TASK_DEFINITION_ARN:-}" ]]; then
       log "Deregistering task definition: $TASK_DEFINITION_ARN"
       if ! aws ecs deregister-task-definition --task-definition
"$TASK_DEFINITION_ARN" 2>>"$LOG_FILE"; then
           log "WARNING: Failed to deregister task definition
$TASK_DEFINITION_ARN"
           ((cleanup_errors++))
       fi
   fi

   # Terminate EC2 instance
   if [[ -n "${INSTANCE_ID:-}" ]]; then
       log "Terminating EC2 instance: $INSTANCE_ID"
       if ! aws ec2 terminate-instances --instance-ids "$INSTANCE_ID"
2>>"$LOG_FILE"; then
           log "WARNING: Failed to terminate instance $INSTANCE_ID"
           ((cleanup_errors++))
       else
           log "Waiting for instance to terminate..."
           if ! aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"
2>>"$LOG_FILE"; then
               log "WARNING: Instance termination wait failed for $INSTANCE_ID"
               ((cleanup_errors++))
           fi
       fi
   fi

   # Delete security group with retry logic
   if [[ -n "${SECURITY_GROUP_ID:-}" ]]; then
       log "Deleting security group: $SECURITY_GROUP_ID"
       local retry_count=0
       local max_retries=3

       while [[ $retry_count -lt $max_retries ]]; do
           if aws ec2 delete-security-group --group-id "$SECURITY_GROUP_ID"
2>>"$LOG_FILE"; then
               log "Successfully deleted security group"
               break
           else
               ((retry_count++))
               if [[ $retry_count -lt $max_retries ]]; then
```

```
                    log "Retry $retry_count/$max_retries: Waiting 10 seconds
 before retrying security group deletion..."
                    sleep 10
                else
                    log "ERROR: Failed to delete security group after
 $max_retries attempts"
                    ((cleanup_errors++))
                fi
            fi
        done
    fi

    # Delete key pair
    if [[ -n "${KEY_PAIR_NAME:-}" ]]; then
        log "Deleting key pair: $KEY_PAIR_NAME"
        if ! aws ec2 delete-key-pair --key-name "$KEY_PAIR_NAME" 2>>"$LOG_FILE";
 then
            log "WARNING: Failed to delete key pair $KEY_PAIR_NAME"
            ((cleanup_errors++))
        fi
        rm -f "${KEY_PAIR_NAME}.pem" 2>>"$LOG_FILE" || log "WARNING: Failed to
 remove local key file"
    fi

    # Delete ECS cluster
    if [[ -n "${CLUSTER_NAME:-}" ]]; then
        log "Deleting ECS cluster: $CLUSTER_NAME"
        if ! aws ecs delete-cluster --cluster "$CLUSTER_NAME" 2>>"$LOG_FILE";
 then
            log "WARNING: Failed to delete cluster $CLUSTER_NAME"
            ((cleanup_errors++))
        fi
    fi

    if [[ $cleanup_errors -eq 0 ]]; then
        log "Cleanup completed successfully"
    else
        log "Cleanup completed with $cleanup_errors warnings/errors. Check log
 file for details."
    fi
}

# Function to check prerequisites
check_prerequisites() {
```

```
    log "Checking prerequisites..."

    # Check AWS CLI
    if ! command -v aws &> /dev/null; then
        log "ERROR: AWS CLI is not installed"
        exit 1
    fi

    # Check AWS credentials
    if ! aws sts get-caller-identity &> /dev/null; then
        log "ERROR: AWS credentials not configured"
        exit 1
    fi

    # Get caller identity
    CALLER_IDENTITY=$(aws sts get-caller-identity --output text --query
 'Account')
    log "AWS Account: $CALLER_IDENTITY"
    log "AWS Region: $AWS_REGION"

    # Check for default VPC
    DEFAULT_VPC=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true"
 --query 'Vpcs[0].VpcId' --output text)
    if [[ "$DEFAULT_VPC" == "None" ]]; then
        log "ERROR: No default VPC found. Please create a VPC first."
        exit 1
    fi
    log "Using default VPC: $DEFAULT_VPC"

    # Get default subnet
    DEFAULT_SUBNET=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=
$DEFAULT_VPC" "Name=default-for-az,Values=true" --query 'Subnets[0].SubnetId' --
output text)
    if [[ "$DEFAULT_SUBNET" == "None" ]]; then
        log "ERROR: No default subnet found"
        exit 1
    fi
    log "Using default subnet: $DEFAULT_SUBNET"

    log "Prerequisites check completed successfully"
}

# Function to create ECS cluster
create_cluster() {
```

```
    log "Creating ECS cluster: $CLUSTER_NAME"

    CLUSTER_ARN=$(aws ecs create-cluster --cluster-name "$CLUSTER_NAME" --
tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG --query
 'cluster.clusterArn' --output text)

    if [[ -z "$CLUSTER_ARN" ]]; then
        log "ERROR: Failed to create cluster"
        exit 1
    fi

    log "Created cluster: $CLUSTER_ARN"
    CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")
}

# Function to create key pair
create_key_pair() {
    log "Creating EC2 key pair: $KEY_PAIR_NAME"

    # FIXED: Set secure umask before key creation
    umask 077
    aws ec2 create-key-pair --key-name "$KEY_PAIR_NAME" --query 'KeyMaterial' --
output text > "${KEY_PAIR_NAME}.pem"
    chmod 400 "${KEY_PAIR_NAME}.pem"
    umask 022  # Reset umask

    log "Created key pair: $KEY_PAIR_NAME"
    CREATED_RESOURCES+=("EC2 Key Pair: $KEY_PAIR_NAME")

    aws ec2 create-tags --resources "$KEY_PAIR_NAME" --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG 2>>"$LOG_FILE" || log "WARNING:
 Failed to tag key pair"
}

# Function to create security group
create_security_group() {
    log "Creating security group: $SECURITY_GROUP_NAME"

    SECURITY_GROUP_ID=$(aws ec2 create-security-group \
        --group-name "$SECURITY_GROUP_NAME" \
        --description "ECS tutorial security group" \
        --vpc-id "$DEFAULT_VPC" \
```

```
        --tag-specifications "ResourceType=security-
group,Tags=[{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=$TUTORIAL_TAG}]"
 \
        --query 'GroupId' --output text)

    if [[ -z "$SECURITY_GROUP_ID" ]]; then
        log "ERROR: Failed to create security group"
        exit 1
    fi

    # Add HTTP access rule for nginx web server with restricted CIDR
    # SECURITY FIX: Restrict access to specific CIDR if available, otherwise
 document the risk
    log "WARNING: Security group allows HTTP (port 80) from 0.0.0.0/0 - restrict
 this in production"
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr "0.0.0.0/0"

    log "Created security group: $SECURITY_GROUP_ID"
    log "Added HTTP access on port 80"
    CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")
}

# Function to get ECS optimized AMI
get_ecs_ami() {
    log "Getting ECS-optimized AMI ID..."

    ECS_AMI_ID=$(aws ssm get-parameters \
        --names /aws/service/ecs/optimized-ami/amazon-linux-2/recommended \
        --query 'Parameters[0].Value' --output text | jq -r '.image_id')

    if [[ -z "$ECS_AMI_ID" ]]; then
        log "ERROR: Failed to get ECS-optimized AMI ID"
        exit 1
    fi

    log "ECS-optimized AMI ID: $ECS_AMI_ID"
}

# Function to create IAM role for ECS instance (if it doesn't exist)
ensure_ecs_instance_role() {
```

```
    log "Checking for ecsInstanceRole..."

    if ! aws iam get-role --role-name ecsInstanceRole &> /dev/null; then
        log "Creating ecsInstanceRole..."

        # Create trust policy
        cat > ecs-instance-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

        # SECURITY FIX: Validate JSON before using
        if ! jq empty ecs-instance-trust-policy.json 2>/dev/null; then
            log "ERROR: Invalid JSON in trust policy"
            rm -f ecs-instance-trust-policy.json
            exit 1
        fi

        # Create role
        aws iam create-role \
            --role-name ecsInstanceRole \
            --assume-role-policy-document file://ecs-instance-trust-policy.json

        aws iam tag-role --role-name ecsInstanceRole --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG

        # Attach managed policy
        aws iam attach-role-policy \
            --role-name ecsInstanceRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role

        # Create instance profile
        aws iam create-instance-profile --instance-profile-name ecsInstanceRole
```

```
        # Add role to instance profile
        aws iam add-role-to-instance-profile \
            --instance-profile-name ecsInstanceRole \
            --role-name ecsInstanceRole

        # FIXED: Enhanced wait for role to be ready
        log "Waiting for IAM role to be ready..."
        aws iam wait role-exists --role-name ecsInstanceRole
        sleep 30  # Additional buffer for eventual consistency

        rm -f ecs-instance-trust-policy.json
        log "Created ecsInstanceRole"
        CREATED_RESOURCES+=("IAM Role: ecsInstanceRole")
    else
        log "ecsInstanceRole already exists"
    fi
}

# Function to launch container instance
launch_container_instance() {
    log "Launching ECS container instance..."

    # Create user data script
    cat > ecs-user-data.sh << EOF
#!/bin/bash
echo ECS_CLUSTER=$CLUSTER_NAME >> /etc/ecs/ecs.config
EOF

    # SECURITY FIX: Validate user data script before use
    if ! bash -n ecs-user-data.sh 2>/dev/null; then
        log "ERROR: Invalid user data script"
        rm -f ecs-user-data.sh
        exit 1
    fi

    INSTANCE_ID=$(aws ec2 run-instances \
        --image-id "$ECS_AMI_ID" \
        --instance-type t3.micro \
        --key-name "$KEY_PAIR_NAME" \
        --security-group-ids "$SECURITY_GROUP_ID" \
        --subnet-id "$DEFAULT_SUBNET" \
        --iam-instance-profile Name=ecsInstanceRole \
        --user-data file://ecs-user-data.sh \
```

```
        --tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=ecs-
tutorial-instance},{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=
$TUTORIAL_TAG}]" \
        --monitoring Enabled=false \
        --metadata-options HttpTokens=required,HttpPutResponseHopLimit=1 \
        --query 'Instances[0].InstanceId' --output text)

    if [[ -z "$INSTANCE_ID" ]]; then
        log "ERROR: Failed to launch EC2 instance"
        rm -f ecs-user-data.sh
        exit 1
    fi

    log "Launched EC2 instance: $INSTANCE_ID"
    log "Instance metadata options: IMDSv2 enforced with hop limit 1"
    CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")

    # Wait for instance to be running
    log "Waiting for instance to be running..."
    aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

    # Wait for ECS agent to register
    log "Waiting for ECS agent to register with cluster..."
    local max_attempts=30
    local attempt=0

    while [[ $attempt -lt $max_attempts ]]; do
        CONTAINER_INSTANCES=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns' --output text)
        if [[ -n "$CONTAINER_INSTANCES" && "$CONTAINER_INSTANCES" != "None" ]];
 then
            log "Container instance registered successfully"
            break
        fi

        attempt=$((attempt + 1))
        log "Waiting for container instance registration... (attempt $attempt/
$max_attempts)"
        sleep 10
    done

    if [[ $attempt -eq $max_attempts ]]; then
        log "ERROR: Container instance failed to register within expected time"
        rm -f ecs-user-data.sh
```

```
        exit 1
    fi

    rm -f ecs-user-data.sh
}

# Function to register task definition
register_task_definition() {
    log "Creating task definition..."

    # Create nginx task definition JSON matching the tutorial
    cat > task-definition.json << 'EOF'
{
    "family": "TASK_FAMILY_PLACEHOLDER",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "cpu": 256,
            "memory": 512,
            "essential": true,
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/nginx-task",
                    "awslogs-region": "REGION_PLACEHOLDER",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ],
    "requiresCompatibilities": ["EC2"],
    "networkMode": "bridge",
    "tags": [
        {
            "key": "project",
            "value": "PROJECT_TAG_PLACEHOLDER"
```

```
        },
        {
            "key": "tutorial",
            "value": "TUTORIAL_TAG_PLACEHOLDER"
        }
    ]
}
EOF

    # Replace placeholders securely
    sed -i "s|TASK_FAMILY_PLACEHOLDER|$TASK_FAMILY|g" task-definition.json
    sed -i "s|REGION_PLACEHOLDER|$AWS_REGION|g" task-definition.json
    sed -i "s|PROJECT_TAG_PLACEHOLDER|$PROJECT_TAG|g" task-definition.json
    sed -i "s|TUTORIAL_TAG_PLACEHOLDER|$TUTORIAL_TAG|g" task-definition.json

    # FIXED: Validate JSON before registration
    if ! jq empty task-definition.json 2>/dev/null; then
        log "ERROR: Invalid JSON in task definition"
        rm -f task-definition.json
        exit 1
    fi

    TASK_DEFINITION_ARN=$(aws ecs register-task-definition \
        --cli-input-json file://task-definition.json \
        --query 'taskDefinition.taskDefinitionArn' --output text)

    if [[ -z "$TASK_DEFINITION_ARN" ]]; then
        log "ERROR: Failed to register task definition"
        rm -f task-definition.json
        exit 1
    fi

    log "Registered task definition: $TASK_DEFINITION_ARN"
    log "Task definition includes CloudWatch Logs configuration for monitoring"
    CREATED_RESOURCES+=("Task Definition: $TASK_DEFINITION_ARN")

    rm -f task-definition.json
}

# Function to create service
create_service() {
    log "Creating ECS service..."

    SERVICE_ARN=$(aws ecs create-service \
```

```
        --cluster "$CLUSTER_NAME" \
        --service-name "$SERVICE_NAME" \
        --task-definition "$TASK_FAMILY" \
        --desired-count 1 \
        --tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG \
        --query 'service.serviceArn' --output text)

    if [[ -z "$SERVICE_ARN" ]]; then
        log "ERROR: Failed to create service"
        exit 1
    fi

    log "Created service: $SERVICE_ARN"
    CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")

    # Wait for service to be stable
    log "Waiting for service to be stable..."
    aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$SERVICE_NAME"

    log "Service is now stable and running"

    # Get the task ARN for monitoring
    TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query 'taskArns[0]' --output text)
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Service task: $TASK_ARN"
        CREATED_RESOURCES+=("ECS Task: $TASK_ARN")
    fi
}

# Function to demonstrate monitoring and testing
demonstrate_monitoring() {
    log "Demonstrating monitoring capabilities..."

    # List services
    log "Listing services in cluster:"
    aws ecs list-services --cluster "$CLUSTER_NAME" --output table

    # Describe service
    log "Service details:"
    aws ecs describe-services --cluster "$CLUSTER_NAME" --
services "$SERVICE_NAME" --output table --query 'services[0].
{ServiceName:serviceName,Status:status,RunningCount:runningCount,DesiredCount:desiredCoun
```

```
    # List tasks
    log "Listing tasks in service:"
    aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name "$SERVICE_NAME"
 --output table

    # Describe task
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Task details:"
        aws ecs describe-tasks --cluster "$CLUSTER_NAME"
 --tasks "$TASK_ARN" --output table --query 'tasks[0].
{TaskArn:taskArn,LastStatus:lastStatus,DesiredStatus:desiredStatus,CreatedAt:createdAt}'
    fi

    # List container instances
    log "Container instances in cluster:"
    aws ecs list-container-instances --cluster "$CLUSTER_NAME" --output table

    # Describe container instance
    CONTAINER_INSTANCE_ARN=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns[0]' --output text)
    if [[ -n "$CONTAINER_INSTANCE_ARN" && "$CONTAINER_INSTANCE_ARN" != "None" ]];
 then
        log "Container instance details:"
        aws ecs describe-container-instances --cluster "$CLUSTER_NAME"
 --container-instances "$CONTAINER_INSTANCE_ARN" --output table --query
 'containerInstances[0].
{Arn:containerInstanceArn,Status:status,RunningTasks:runningTasksCount,PendingTasks:pendi
    fi

    # Test the nginx web server
    log "Testing nginx web server..."
    PUBLIC_IP=$(aws ec2 describe-instances --instance-ids "$INSTANCE_ID" --query
 'Reservations[0].Instances[0].PublicIpAddress' --output text)

    if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
        log "Container instance public IP: $PUBLIC_IP"
        log "Testing HTTP connection to nginx..."

        # Wait a moment for nginx to be fully ready
        sleep 10

        if curl -s --connect-timeout 10 "http://$PUBLIC_IP" | grep -q "Welcome to
 nginx"; then
```

```
            log "SUCCESS: Nginx web server is responding correctly"
            echo ""
            echo "=========================================="
            echo "WEB SERVER TEST SUCCESSFUL"
            echo "=========================================="
            echo "You can access your nginx web server at: http://$PUBLIC_IP"
            echo "The nginx welcome page should be visible in your browser."
        else
            log "WARNING: Nginx web server may not be fully ready yet. Try
 accessing http://$PUBLIC_IP in a few minutes."
        fi
    else
        log "WARNING: Could not retrieve public IP address"
    fi
}

# Main execution
main() {
    log "Starting ECS EC2 Launch Type Tutorial (UPDATED VERSION)"
    log "Log file: $LOG_FILE"
    log "Security improvements: IMDSv2 enforced, JSON validation, input
 sanitization, CloudWatch Logs configured"

    check_prerequisites
    create_cluster
    create_key_pair
    create_security_group
    get_ecs_ami
    ensure_ecs_instance_role
    launch_container_instance
    register_task_definition
    create_service
    demonstrate_monitoring

    log "Tutorial completed successfully!"

    echo ""
    echo "=========================================="
    echo "TUTORIAL COMPLETED SUCCESSFULLY"
    echo "=========================================="
    echo "Resources created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
```

```
    echo ""
    echo "The nginx service will continue running and maintain the desired task
 count."
    echo "You can monitor the service status using:"
    echo "  aws ecs describe-services --cluster $CLUSTER_NAME --services
 $SERVICE_NAME"
    echo ""
    if [[ -n "${PUBLIC_IP:-}" ]]; then
        echo "Access your web server at: http://$PUBLIC_IP"
        echo ""
    fi
    echo "============================================="
    echo "CLEANUP CONFIRMATION"
    echo "============================================="
    log "Auto-confirming cleanup - proceeding with resource cleanup"
    cleanup_resources
    log "All resources have been cleaned up"
}

# Run main function
main "$@"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateInstanceProfile
  - CreateKeyPair
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DeleteService

- DeregisterTaskDefinition
- DescribeContainerInstances
- DescribeInstances
- DescribeServices
- DescribeSubnets
- DescribeTasks
- DescribeVpcs
- GetCallerIdentity
- GetParameters
- GetRole
- ListContainerInstances
- ListServices
- ListTasks
- RegisterTaskDefinition
- RunInstances
- StopTask
- TerminateInstances
- UpdateService
- Attendre (Amazon EC2)
- Attendre (Amazon ECS)
- Attendre (IAM)

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Création d'un espace de travail de surveillance géré

L'exemple de code suivant illustre comment :

- Créez un rôle IAM pour votre espace de travail
- Création d'un espace de travail Grafana
- Configuration de l'authentification

- Configurer les paramètres facultatifs
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Amazon Managed Grafana Workspace Creation Script
# This script creates an Amazon Managed Grafana workspace and configures it

# Set up logging
LOG_FILE="grafana-workspace-creation.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Managed Grafana workspace creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -i "error\|exception\|fail" > /dev/null; then
        echo "ERROR: Command '$cmd' failed with output:"
        echo "$output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Attempting to clean up resources..."
```

```
    if [ -n "$WORKSPACE_ID" ]; then
        echo "Deleting workspace $WORKSPACE_ID..."
        aws grafana delete-workspace --workspace-id "$WORKSPACE_ID"
    fi

    if [ -n "$ROLE_NAME" ]; then
        echo "Detaching policies from role $ROLE_NAME..."
        if [ -n "$POLICY_ARN" ]; then
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$POLICY_ARN"
        fi

        echo "Deleting role $ROLE_NAME..."
        aws iam delete-role --role-name "$ROLE_NAME"
    fi

    if [ -n "$POLICY_ARN" ]; then
        echo "Deleting policy..."
        aws iam delete-policy --policy-arn "$POLICY_ARN"
    fi

    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json

    echo "Cleanup completed. See $LOG_FILE for details."
}

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
WORKSPACE_NAME="GrafanaWorkspace-${RANDOM_ID}"
ROLE_NAME="GrafanaWorkspaceRole-${RANDOM_ID}"

echo "Using workspace name: $WORKSPACE_NAME"
echo "Using role name: $ROLE_NAME"

# Step 1: Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
check_error "$ACCOUNT_ID" "get-caller-identity"
echo "AWS Account ID: $ACCOUNT_ID"

# Step 2: Create IAM role for Grafana workspace
echo "Creating IAM role for Grafana workspace..."
```

```
# Create trust policy document
cat > trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "grafana.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role
ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
  --assume-role-policy-document file://trust-policy.json \
  --description "Role for Amazon Managed Grafana workspace")

check_error "$ROLE_OUTPUT" "create-role"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-grafana-gs
echo "IAM role created successfully"

# Extract role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "Role ARN: $ROLE_ARN"

# Attach policies to the role
echo "Attaching policies to the role..."

# CloudWatch policy
cat > cloudwatch-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:DescribeAlarmsForMetric",
```

```
        "cloudwatch:DescribeAlarmHistory",
        "cloudwatch:DescribeAlarms",
        "cloudwatch:ListMetrics",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:GetMetricData"
      ],
      "Resource": "*"
    }
  ]
}
EOF

POLICY_OUTPUT=$(aws iam create-policy \
  --policy-name "GrafanaCloudWatchPolicy-${RANDOM_ID}" \
  --policy-document file://cloudwatch-policy.json)

check_error "$POLICY_OUTPUT" "create-policy"

POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "CloudWatch policy ARN: $POLICY_ARN"

ATTACH_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-arn "$POLICY_ARN")

check_error "$ATTACH_OUTPUT" "attach-role-policy"
echo "CloudWatch policy attached to role"

# Step 3: Create the Grafana workspace
echo "Creating Amazon Managed Grafana workspace..."
WORKSPACE_OUTPUT=$(aws grafana create-workspace \
  --workspace-name "$WORKSPACE_NAME" \
  --authentication-providers "SAML" \
  --permission-type "CUSTOMER_MANAGED" \
  --account-access-type "CURRENT_ACCOUNT" \
  --workspace-role-arn "$ROLE_ARN" \
  --workspace-data-sources "CLOUDWATCH" "PROMETHEUS" "XRAY" \
  --grafana-version "10.4" \
  --tags Environment=Development,project=doc-smith,tutorial=amazon-managed-
grafana-gs)

check_error "$WORKSPACE_OUTPUT" "create-workspace"

echo "Workspace creation initiated:"
```

```
echo "$WORKSPACE_OUTPUT"

# Extract workspace ID
WORKSPACE_ID=$(echo "$WORKSPACE_OUTPUT" | grep -o '"id": "[^"]*' | cut -d'"' -f4)

if [ -z "$WORKSPACE_ID" ]; then
    echo "ERROR: Failed to extract workspace ID from output"
    exit 1
fi

echo "Workspace ID: $WORKSPACE_ID"

# Step 4: Wait for workspace to become active
echo "Waiting for workspace to become active. This may take several minutes..."
ACTIVE=false
MAX_ATTEMPTS=30
ATTEMPT=0

while [ $ACTIVE = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
    ATTEMPT=$((ATTEMPT+1))
    echo "Checking workspace status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."

    DESCRIBE_OUTPUT=$(aws grafana describe-workspace --workspace-id
 "$WORKSPACE_ID")
    check_error "$DESCRIBE_OUTPUT" "describe-workspace"

    STATUS=$(echo "$DESCRIBE_OUTPUT" | grep -o '"status": "[^"]*' | cut -d'"' -
f4)
    echo "Current status: $STATUS"

    if [ "$STATUS" = "ACTIVE" ]; then
        ACTIVE=true
        echo "Workspace is now ACTIVE"
    elif [ "$STATUS" = "FAILED" ]; then
        echo "ERROR: Workspace creation failed"
        cleanup_on_error
        exit 1
    else
        echo "Workspace is still being created. Waiting 30 seconds..."
        sleep 30
    fi
done

if [ $ACTIVE = false ]; then
```

```
    echo "ERROR: Workspace did not become active within the expected time"
    cleanup_on_error
    exit 1
fi

# Extract workspace endpoint URL
WORKSPACE_URL=$(echo "$DESCRIBE_OUTPUT" | grep -o '"endpoint": "[^"]*' | cut -
d'"' -f4)
echo "Workspace URL: https://$WORKSPACE_URL"

# Step 5: Display workspace information
echo ""
echo "==========================================="
echo "WORKSPACE INFORMATION"
echo "==========================================="
echo "Workspace ID: $WORKSPACE_ID"
echo "Workspace URL: https://$WORKSPACE_URL"
echo "Workspace Name: $WORKSPACE_NAME"
echo "IAM Role: $ROLE_NAME"
echo ""
echo "Note: Since SAML authentication is used, you need to configure SAML
 settings"
echo "using the AWS Management Console or the update-workspace-authentication
 command."
echo "==========================================="

# Step 6: Prompt for cleanup
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Resources created:"
echo "- Amazon Managed Grafana workspace: $WORKSPACE_ID"
echo "- IAM Role: $ROLE_NAME"
echo "- IAM Policy: GrafanaCloudWatchPolicy-${RANDOM_ID}"
echo ""
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy] ]]; then
    echo "Cleaning up resources..."

    echo "Deleting workspace $WORKSPACE_ID..."
    DELETE_OUTPUT=$(aws grafana delete-workspace --workspace-id "$WORKSPACE_ID")
```

```
    check_error "$DELETE_OUTPUT" "delete-workspace"

    echo "Waiting for workspace to be deleted..."
    DELETED=false
    ATTEMPT=0

    while [ $DELETED = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
        ATTEMPT=$((ATTEMPT+1))
        echo "Checking deletion status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."

        if aws grafana describe-workspace --workspace-id "$WORKSPACE_ID" 2>&1 |
grep -i "not found\|does not exist" > /dev/null; then
            DELETED=true
            echo "Workspace has been deleted"
        else
            echo "Workspace is still being deleted. Waiting 30 seconds..."
            sleep 30
        fi
    done

    if [ $DELETED = false ]; then
        echo "WARNING: Workspace deletion is taking longer than expected. It may
still be in progress."
    fi

    # Detach policy from role
    echo "Detaching policy from role..."
    aws iam detach-role-policy \
      --role-name "$ROLE_NAME" \
      --policy-arn "$POLICY_ARN"

    # Delete policy
    echo "Deleting IAM policy..."
    aws iam delete-policy \
      --policy-arn "$POLICY_ARN"

    # Delete role
    echo "Deleting IAM role..."
    aws iam delete-role \
      --role-name "$ROLE_NAME"

    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json
```

```
    echo "Cleanup completed"
else
    echo "Skipping cleanup. Resources will remain in your AWS account."
fi

echo "Script completed at $(date)"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreatePolicy
  - CreateRole
  - CreateWorkspace
  - DeletePolicy
  - DeleteRole
  - DeleteWorkspace
  - DescribeWorkspace
  - DetachRolePolicy
  - GetCallerIdentity

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Création d'un tableau de bord de surveillance avec le nom de la fonction comme variable

L'exemple de code suivant illustre comment :

- Création d'un CloudWatch tableau de bord
- Ajouter des widgets de métriques Lambda avec une variable de nom de fonction
- Vérifiez le tableau de bord
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Script to create a CloudWatch dashboard with Lambda function name as a variable
# This script creates a CloudWatch dashboard that allows you to switch between
 different Lambda functions

# Set up logging with secure permissions
LOG_FILE="${HOME}/.cloudwatch-dashboard-script.log"
touch "$LOG_FILE" && chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "$(date): Starting CloudWatch dashboard creation script"

# Security: Set strict error handling
set -uo pipefail
trap 'handle_error "Script failed at line $LINENO"' ERR

# Function to handle errors
handle_error() {
    local error_msg="${1:-Unknown error}"
    echo "ERROR: $error_msg" >&2
    echo "Resources created:"
    echo "- CloudWatch Dashboard: LambdaMetricsDashboard"
    echo ""
    echo "=========================================="
    echo "CLEANUP CONFIRMATION"
    echo "=========================================="
    echo "An error occurred. Proceeding with automatic cleanup..."

    echo "Cleaning up resources..."
    aws cloudwatch delete-dashboards --dashboard-names LambdaMetricsDashboard 2>/
dev/null || true
```

```
    # Clean up temporary files
    if [ -n "${TEMP_DIR:-}" ] && [ -d "$TEMP_DIR" ]; then
        rm -rf "$TEMP_DIR"
    fi
    rm -f dashboard-body.json

    echo "Cleanup complete."
    exit 1
}

# Security: Validate AWS CLI is installed
if ! command -v aws &> /dev/null; then
    handle_error "AWS CLI is not installed. Please install it and try again."
fi

# Check if AWS CLI is installed and configured
echo "Checking AWS CLI configuration..."
if ! aws sts get-caller-identity > /dev/null 2>&1; then
    handle_error "AWS CLI is not properly configured. Please configure it with
 'aws configure' and try again."
fi

# Get the current region securely
REGION=$(aws configure get region 2>/dev/null || echo "")
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $REGION"
fi
echo "Using region: $REGION"

# Validate region format
if ! [[ "$REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]{1}$ ]]; then
    handle_error "Invalid AWS region format: $REGION"
fi

# Check if there are any Lambda functions in the account
echo "Checking for Lambda functions..."
LAMBDA_FUNCTIONS=$(aws lambda list-functions --region "$REGION" --query
 "Functions[*].FunctionName" --output text 2>/dev/null || echo "")

if [ -z "$LAMBDA_FUNCTIONS" ]; then
    echo "No Lambda functions found in your account. Creating a simple test
 function..."
```

```
    # Create a temporary directory for Lambda function code with secure
 permissions
    TEMP_DIR=$(mktemp -d)
    chmod 700 "$TEMP_DIR"
    trap 'rm -rf "$TEMP_DIR"' EXIT

    # Create a simple Lambda function
    cat > "$TEMP_DIR/index.js" << 'EOF'
exports.handler = async (event) => {
    console.log('Event:', JSON.stringify(event, null, 2));
    return {
        statusCode: 200,
        body: JSON.stringify('Hello from Lambda!'),
    };
};
EOF

    # Zip the function code
    if ! cd "$TEMP_DIR"; then
        handle_error "Failed to change to temporary directory"
    fi

    if ! zip -q function.zip index.js; then
        handle_error "Failed to create zip file"
    fi

    # Create a role for the Lambda function with restricted trust policy
    ROLE_NAME="LambdaDashboardTestRole-$(date +%s)"
    TRUST_POLICY='{"Version":"2012-10-17",       "Statement":
[{"Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}'

    if ! ROLE_ARN=$(aws iam create-role \
        --role-name "$ROLE_NAME" \
        --assume-role-policy-document "$TRUST_POLICY" \
        --tags Key=project,Value=doc-smith Key=tutorial,Value=cloudwatch-
dynamicdash \
        --query "Role.Arn" \
        --output text 2>/dev/null); then
        handle_error "Failed to create IAM role for Lambda function"
    fi

    echo "Waiting for role to be available..."
```

```
    sleep 10

    # Attach basic Lambda execution policy
    if ! aws iam attach-role-policy \
        --role-name "$ROLE_NAME" \
        --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole"; then
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to attach policy to IAM role"
    fi

    # Create the Lambda function
    FUNCTION_NAME="DashboardTestFunction-$(date +%s)"
    if ! aws lambda create-function \
        --function-name "$FUNCTION_NAME" \
        --runtime nodejs18.x \
        --role "$ROLE_ARN" \
        --handler index.handler \
        --zip-file fileb://function.zip \
        --tags project=doc-smith,tutorial=cloudwatch-dynamicdash \
        --region "$REGION" > /dev/null 2>&1; then
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to create Lambda function"
    fi

    # Invoke the function to generate some metrics
    echo "Invoking Lambda function to generate metrics..."
    for i in {1..5}; do
        aws lambda invoke --function-name "$FUNCTION_NAME" --payload '{}' /dev/
null --region "$REGION" > /dev/null 2>&1 || true
        sleep 1
    done

    # Go back to original directory
    cd - > /dev/null

    # Set the function name for the dashboard
    DEFAULT_FUNCTION="$FUNCTION_NAME"
else
    # Use the first Lambda function as default
```

```
    DEFAULT_FUNCTION=$(echo "$LAMBDA_FUNCTIONS" | awk '{print $1}')
    echo "Found Lambda functions. Using $DEFAULT_FUNCTION as default."
    FUNCTION_NAME=""
    ROLE_NAME=""
fi

# Create a dashboard with Lambda metrics and a function name variable
echo "Creating CloudWatch dashboard with Lambda function name variable..."

# Create a JSON file for the dashboard body with secure permissions
DASHBOARD_JSON="dashboard-body-$$.json"
touch "$DASHBOARD_JSON" && chmod 600 "$DASHBOARD_JSON"

# Escape special characters in region and function name for JSON
REGION_ESCAPED=$(printf '%s\n' "$REGION" | sed 's:[\/&]:\\&:g')
FUNCTION_ESCAPED=$(printf '%s\n' "$DEFAULT_FUNCTION" | sed 's:[\/&]:\\&:g')

cat > "$DASHBOARD_JSON" << EOF
{
  "widgets": [
    {
      "type": "metric",
      "x": 0,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Invocations", "FunctionName", "\${FunctionName}" ],
          [ ".", "Errors", ".", "." ],
          [ ".", "Throttles", ".", "." ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Lambda Function Metrics for \${FunctionName}",
        "period": 300
      }
    },
    {
      "type": "metric",
      "x": 0,
      "y": 6,
      "width": 12,
```

```
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Duration", "FunctionName", "\${FunctionName}",
 { "stat": "Average" } ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Duration for \${FunctionName}",
        "period": 300
      }
    },
    {
      "type": "metric",
      "x": 12,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "ConcurrentExecutions", "FunctionName", "\
${FunctionName}" ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Concurrent Executions for \${FunctionName}",
        "period": 300
      }
    }
  ],
  "periodOverride": "auto",
  "variables": [
    {
      "type": "property",
      "id": "FunctionName",
      "property": "FunctionName",
      "label": "Lambda Function",
      "inputType": "select",
      "values": [
        {
          "value": "$FUNCTION_ESCAPED",
          "label": "$FUNCTION_ESCAPED"
```

```
        }
      ]
    }
  ]
}
EOF

# Validate JSON before sending
if ! jq empty "$DASHBOARD_JSON" 2>/dev/null; then
    handle_error "Invalid JSON generated for dashboard"
fi

# Create the dashboard using the JSON file
DASHBOARD_NAME="LambdaMetricsDashboard-$(date +%s)"
if ! DASHBOARD_RESULT=$(aws cloudwatch put-dashboard \
    --dashboard-name "$DASHBOARD_NAME" \
    --dashboard-body file://"$DASHBOARD_JSON" \
    --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to create CloudWatch dashboard."
fi

# Display any validation messages but continue
if echo "$DASHBOARD_RESULT" | grep -q "DashboardValidationMessages"; then
    echo "Dashboard created with validation messages:"
    echo "$DASHBOARD_RESULT"
    echo "These validation messages are warnings and the dashboard should still
 function."
else
    echo "Dashboard created successfully!"
fi

# Verify the dashboard was created
echo "Verifying dashboard creation..."
```

```
if ! DASHBOARD_INFO=$(aws cloudwatch get-dashboard --dashboard-name
 "$DASHBOARD_NAME" --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to verify dashboard creation."
fi

echo "Dashboard verification successful!"
echo "Dashboard details:"
echo "$DASHBOARD_INFO"

# List all dashboards to confirm
echo "Listing all dashboards:"
if ! DASHBOARDS=$(aws cloudwatch list-dashboards --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to list dashboards."
fi
echo "$DASHBOARDS"

# Show instructions for accessing the dashboard
echo ""
echo "Dashboard created successfully! To access it:"
echo "1. Open the CloudWatch console at https://console.aws.amazon.com/
cloudwatch/"
echo "2. In the navigation pane, choose Dashboards"
echo "3. Select $DASHBOARD_NAME"
```

```
echo "4. You should see a dropdown menu labeled 'Lambda Function' at the top of
 the dashboard"
echo "5. Use this dropdown to select different Lambda functions and see their
 metrics"
echo ""

# Create a list of resources for cleanup
RESOURCES=("- CloudWatch Dashboard: $DASHBOARD_NAME")
if [ -n "${FUNCTION_NAME:-}" ]; then
    RESOURCES+=("- Lambda Function: $FUNCTION_NAME")
    RESOURCES+=("- IAM Role: $ROLE_NAME")
fi

# Prompt for cleanup with automatic yes
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Resources created:"
for resource in "${RESOURCES[@]}"; do
    echo "$resource"
done
echo ""
echo "Proceeding with automatic cleanup..."

CLEANUP_CHOICE="y"

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."

    # Delete the dashboard
    if aws cloudwatch delete-dashboards --dashboard-names "$DASHBOARD_NAME" --
region "$REGION" 2>/dev/null; then
        echo "Dashboard deleted successfully."
    else
        echo "WARNING: Failed to delete dashboard. You may need to delete it
 manually."
    fi

    # If we created a Lambda function, delete it and its role
    if [ -n "${FUNCTION_NAME:-}" ]; then
        echo "Deleting Lambda function..."
        if aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null; then
            echo "Lambda function deleted successfully."
```

```
        else
            echo "WARNING: Failed to delete Lambda function. You may need to
 delete it manually."
        fi

        echo "Detaching role policy..."
        if aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null; then
            echo "Role policy detached successfully."
        else
            echo "WARNING: Failed to detach role policy. You may need to detach
 it manually."
        fi

        echo "Deleting IAM role..."
        if aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null; then
            echo "IAM role deleted successfully."
        else
            echo "WARNING: Failed to delete IAM role. You may need to delete it
 manually."
        fi
    fi

    # Clean up the JSON file
    rm -f "$DASHBOARD_JSON"

    echo "Cleanup complete."
fi

echo "Script completed successfully!"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteDashboards
  - DeleteFunction
  - DeleteRole

- DetachRolePolicy
- GetCallerIdentity
- GetDashboard
- Invoke
- ListDashboards
- ListFunctions
- PutDashboard

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Création de votre première fonction sans serveur

L'exemple de code suivant illustre comment :

- Création d'un rôle IAM pour Lambda
- Créer un code de fonction
- Création d'une fonction Lambda
- Testez votre fonction Lambda
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash
# AWS Lambda - Create Your First Function
# This script creates a Lambda function, invokes it with a test event,
# views CloudWatch logs, and cleans up all resources.
```

```
#
# Source: https://docs.aws.amazon.com/lambda/latest/dg/getting-started.html
#
# Resources created:
#   - IAM role (Lambda execution role with basic logging permissions)
#   - Lambda function (Python 3.13 or Node.js 22.x runtime)
#   - CloudWatch log group (created automatically by Lambda on invocation)

set -eE -o pipefail

###############################################################################
# Setup
###############################################################################

UNIQUE_ID=$(head -c 8 /dev/urandom | od -An -tx1 | tr -d ' ')
FUNCTION_NAME="my-lambda-function-${UNIQUE_ID}"
ROLE_NAME="lambda-execution-role-${UNIQUE_ID}"
LOG_GROUP_NAME="/aws/lambda/${FUNCTION_NAME}"

TEMP_DIR=$(mktemp -d)
readonly TEMP_DIR
LOG_FILE="${TEMP_DIR}/lambda-gettingstarted.log"

exec > >(tee -a "$LOG_FILE") 2>&1

declare -a CREATED_RESOURCES

###############################################################################
# Helper functions
###############################################################################

cleanup_resources() {
    # Disable error trap to prevent recursion during cleanup
    trap - ERR
    set +eE

    echo ""
    echo "Cleaning up resources..."
    echo ""

    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        local RESOURCE="${CREATED_RESOURCES[$i]}"
        local TYPE="${RESOURCE%%:*}"
        local NAME="${RESOURCE#*:}"
```

```
        case "$TYPE" in
            log-group)
                echo "Deleting CloudWatch log group: ${NAME}"
                aws logs delete-log-group \
                    --log-group-name "$NAME" 2>&1 || echo "  WARNING: Could not
delete log group ${NAME}."
                ;;
            lambda-function)
                echo "Deleting Lambda function: ${NAME}"
                aws lambda delete-function \
                    --function-name "$NAME" 2>&1 || echo "  WARNING: Could not
delete Lambda function ${NAME}."
                echo "  Waiting for function deletion to complete..."
                local DELETE_WAIT=0
                while aws lambda get-function --function-name "$NAME" > /dev/null
2>&1; do
                    sleep 2
                    DELETE_WAIT=$((DELETE_WAIT + 2))
                    if [ "$DELETE_WAIT" -ge 60 ]; then
                        echo "  WARNING: Timed out waiting for function
deletion."
                        break
                    fi
                done
                ;;
            iam-role-policy)
                local ROLE_PART="${NAME%%|*}"
                local POLICY_PART="${NAME#*|}"
                echo "Detaching policy from role: ${ROLE_PART}"
                aws iam detach-role-policy \
                    --role-name "$ROLE_PART" \
                    --policy-arn "$POLICY_PART" 2>&1 || echo "  WARNING: Could
not detach policy from role ${ROLE_PART}."
                ;;
            iam-role)
                echo "Deleting IAM role: ${NAME}"
                aws iam delete-role \
                    --role-name "$NAME" 2>&1 || echo "  WARNING: Could not delete
IAM role ${NAME}."
                ;;
        esac
    done
```

```
    if [ -d "$TEMP_DIR" ]; then
        rm -rf "$TEMP_DIR"
    fi

    echo ""
    echo "Cleanup complete."
}

handle_error() {
    echo ""
    echo "========================================="
    echo "ERROR: Script failed at $1"
    echo "========================================="
    echo ""
    if [ ${#CREATED_RESOURCES[@]} -gt 0 ]; then
        echo "Attempting to clean up ${#CREATED_RESOURCES[@]} resource(s)..."
        cleanup_resources
    fi
    exit 1
}

trap 'handle_error "line $LINENO"' ERR

wait_for_resource() {
    local DESCRIPTION="$1"
    local COMMAND="$2"
    local TARGET_VALUE="$3"
    local TIMEOUT=300
    local ELAPSED=0
    local INTERVAL=5

    echo "Waiting for ${DESCRIPTION}..."
    while true; do
        local RESULT
        RESULT=$(eval "$COMMAND" 2>&1) || true
        if echo "$RESULT" | grep -q "$TARGET_VALUE"; then
            echo "  ${DESCRIPTION} is ready."
            return 0
        fi
        if [ "$ELAPSED" -ge "$TIMEOUT" ]; then
            echo "ERROR: Timed out waiting for ${DESCRIPTION} after ${TIMEOUT}
 seconds."
            return 1
        fi
```

```
        sleep "$INTERVAL"
        ELAPSED=$((ELAPSED + INTERVAL))
    done
}

validate_input() {
    local input="$1"
    local pattern="$2"
    if ! [[ "$input" =~ $pattern ]]; then
        echo "ERROR: Invalid input: $input"
        return 1
    fi
    return 0
}

# ============================================================================
# Region pre-check
# ============================================================================

CONFIGURED_REGION=$(aws configure get region 2>/dev/null || true)
if [ -z "$CONFIGURED_REGION" ] && [ -z "$AWS_DEFAULT_REGION" ] && [ -z
 "$AWS_REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Run 'aws configure set region <region>' or export AWS_DEFAULT_REGION."
    exit 1
fi

# ============================================================================
# Runtime selection
# ============================================================================

echo ""
echo "=========================================="
echo "AWS Lambda - Create Your First Function"
echo "=========================================="
echo ""
echo "Select a runtime for your Lambda function:"
echo "  1) Python 3.13"
echo "  2) Node.js 22.x"
echo ""
echo "Using default: Python 3.13"
RUNTIME_CHOICE="1"

case "$RUNTIME_CHOICE" in
```

```
    1)
        RUNTIME="python3.13"
        HANDLER="lambda_function.lambda_handler"
        CODE_FILE="lambda_function.py"
        cat > "${TEMP_DIR}/${CODE_FILE}" << 'PYTHON_EOF'
import json
import logging
logger = logging.getLogger()
logger.setLevel(logging.INFO)
def lambda_handler(event, context):
    if not isinstance(event, dict) or 'length' not in event or 'width' not in
 event:
        raise ValueError('Event must contain length and width')
    try:
        length = float(event['length'])
        width = float(event['width'])
        if length < 0 or width < 0:
            raise ValueError('Length and width must be non-negative')
        area = calculate_area(length, width)
        print(f'The area is {area}')
        logger.info(f'CloudWatch logs group: {context.log_group_name}')
        return json.dumps({'area': area})
    except (TypeError, ValueError) as e:
        logger.error(f'Error processing input: {str(e)}')
        raise
def calculate_area(length, width):
    return length * width
PYTHON_EOF
        echo "Selected runtime: Python 3.13"
        ;;
    2)
        RUNTIME="nodejs22.x"
        HANDLER="index.handler"
        CODE_FILE="index.mjs"
        cat > "${TEMP_DIR}/${CODE_FILE}" << 'NODEJS_EOF'
export const handler = async (event, context) => {
  if (!event || typeof event.length !== 'number' || typeof event.width !==
 'number') {
    throw new Error('Event must contain numeric length and width');
  }
  if (event.length < 0 || event.width < 0) {
    throw new Error('Length and width must be non-negative');
  }
  const area = event.length * event.width;
```

```
  console.log(`The area is ${area}`);
  console.log('CloudWatch log group: ', context.logGroupName);
  return JSON.stringify({area});
};
NODEJS_EOF
        echo "Selected runtime: Node.js 22.x"
        ;;
    *)
        echo "ERROR: Invalid choice. Please enter 1 or 2."
        exit 1
        ;;
esac

# ============================================================================
# Step 1: Create IAM execution role
# ============================================================================

echo ""
echo "==========================================="
echo "Step 1: Create IAM execution role"
echo "==========================================="
echo ""

TRUST_POLICY='{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "lambda.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}'

echo "Creating IAM role: ${ROLE_NAME}"
ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document "$TRUST_POLICY" \
    --query 'Role.Arn' \
    --output text 2>&1)

if ! validate_input "$ROLE_OUTPUT" "^arn:aws:iam::[0-9]+:role/"; then
```

```
    echo "ERROR: Failed to create IAM role"
    exit 1
fi

echo "$ROLE_OUTPUT"
ROLE_ARN="$ROLE_OUTPUT"
CREATED_RESOURCES+=("iam-role:${ROLE_NAME}")
echo "Role ARN: ${ROLE_ARN}"

aws iam tag-role \
    --role-name "$ROLE_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=lambda-gettingstarted

echo ""
echo "Attaching AWSLambdaBasicExecutionRole policy..."
aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>&1
CREATED_RESOURCES+=("iam-role-policy:${ROLE_NAME}|arn:aws:iam::aws:policy/
service-role/AWSLambdaBasicExecutionRole")
echo "Policy attached."

# IAM roles can take a few seconds to propagate
echo "Waiting for IAM role to propagate..."
sleep 10

################################################################################
# Step 2: Create Lambda function
################################################################################

echo ""
echo "========================================="
echo "Step 2: Create Lambda function"
echo "========================================="
echo ""

echo "Creating deployment package..."
ORIGINAL_DIR=$(pwd)
cd "$TEMP_DIR" || exit 1
zip -j function.zip "$CODE_FILE" > /dev/null 2>&1 || {
    echo "ERROR: Failed to create deployment package"
    exit 1
}
```

```
cd "$ORIGINAL_DIR" || exit 1

if [ ! -f "${TEMP_DIR}/function.zip" ]; then
    echo "ERROR: Deployment package creation failed"
    exit 1
fi

echo "Creating Lambda function: ${FUNCTION_NAME}"
echo "  Runtime: ${RUNTIME}"
echo "  Handler: ${HANDLER}"
echo ""

CREATE_OUTPUT=$(aws lambda create-function \
    --function-name "$FUNCTION_NAME" \
    --runtime "$RUNTIME" \
    --role "$ROLE_ARN" \
    --handler "$HANDLER" \
    --architectures x86_64 \
    --zip-file "fileb://${TEMP_DIR}/function.zip" \
    --tags project=doc-smith,tutorial=lambda-gettingstarted \
    --query '[FunctionName, FunctionArn, Runtime, State]' \
    --output text 2>&1)

if [ -z "$CREATE_OUTPUT" ]; then
    echo "ERROR: Failed to create Lambda function"
    exit 1
fi

echo "$CREATE_OUTPUT"
CREATED_RESOURCES+=("lambda-function:${FUNCTION_NAME}")

wait_for_resource "Lambda function to become Active" \
    "aws lambda get-function-configuration --function-name ${FUNCTION_NAME} --
query State --output text" \
    "Active"

################################################################################
# Step 3: Invoke the function
################################################################################

echo ""
echo "============================================"
echo "Step 3: Invoke the function"
echo "============================================"
```

```
echo ""

TEST_EVENT='{"length": 6, "width": 7}'
echo "Invoking function with test event: ${TEST_EVENT}"
echo ""

echo "$TEST_EVENT" > "${TEMP_DIR}/test-event.json"

if ! validate_input "$TEST_EVENT" '"length": [0-9]+, "width": [0-9]+'; then
    echo "ERROR: Invalid test event format"
    exit 1
fi

INVOKE_OUTPUT=$(aws lambda invoke \
    --function-name "$FUNCTION_NAME" \
    --payload "fileb://${TEMP_DIR}/test-event.json" \
    --cli-read-timeout 30 \
    "${TEMP_DIR}/response.json" 2>&1)
echo "$INVOKE_OUTPUT"

if [ ! -f "${TEMP_DIR}/response.json" ]; then
    echo "ERROR: No response file generated"
    exit 1
fi

RESPONSE=$(cat "${TEMP_DIR}/response.json")
echo ""
echo "Function response: ${RESPONSE}"
echo ""

if echo "$INVOKE_OUTPUT" | grep -qi "functionerror"; then
    echo "WARNING: Function returned an error."
fi

################################################################################
# Step 4: View CloudWatch logs
################################################################################

echo ""
echo "========================================="
echo "Step 4: View CloudWatch Logs"
echo "========================================="
echo ""
```

```
echo "Log group: ${LOG_GROUP_NAME}"
echo ""

echo "Waiting for CloudWatch logs to be available..."

LOG_STREAMS=""
for i in $(seq 1 6); do
    LOG_STREAMS=$(aws logs describe-log-streams \
        --log-group-name "$LOG_GROUP_NAME" \
        --order-by LastEventTime \
        --descending \
        --query 'logStreams[0].logStreamName' \
        --output text 2>/dev/null) || true
    if [ -n "$LOG_STREAMS" ] && [ "$LOG_STREAMS" != "None" ]; then
        break
    fi
    LOG_STREAMS=""
    sleep 5
done

if [ -n "$LOG_STREAMS" ] && [ "$LOG_STREAMS" != "None" ]; then
    echo "Latest log stream: ${LOG_STREAMS}"
    echo ""
    echo "--- Log events ---"
    LOG_EVENTS=$(aws logs get-log-events \
        --log-group-name "$LOG_GROUP_NAME" \
        --log-stream-name "$LOG_STREAMS" \
        --query 'events[].message' \
        --output text 2>&1) || true
    echo "$LOG_EVENTS"
    echo "--- End of log events ---"
else
    echo "No log streams found yet. Logs may take a moment to appear."
    echo "You can view them in the CloudWatch console:"
    echo "  Log group: ${LOG_GROUP_NAME}"
fi

CREATED_RESOURCES+=("log-group:${LOG_GROUP_NAME}")

aws logs tag-log-group \
    --log-group-name "$LOG_GROUP_NAME" \
    --tags project=doc-smith,tutorial=lambda-gettingstarted

################################################################################
```

```
# Summary and cleanup
################################################################################

echo ""
echo "==========================================="
echo "SUMMARY"
echo "==========================================="
echo ""
echo "Resources created:"
echo "  IAM role:          ${ROLE_NAME}"
echo "  Lambda function:   ${FUNCTION_NAME}"
echo "  CloudWatch logs:   ${LOG_GROUP_NAME}"
echo ""
echo "==========================================="
echo "CLEANUP"
echo "==========================================="
echo ""
echo "Cleaning up all created resources..."
cleanup_resources

echo ""
echo "Done."
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteFunction
  - DeleteLogGroup
  - DeleteRole
  - DescribeLogStreams
  - DetachRolePolicy
  - GetFunction
  - GetFunctionConfiguration
  - GetLogEvents
  - Invoke

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Commencez avec les entrepôts de données sans serveur à l'aide de la CLI

L'exemple de code suivant illustre comment :

- Utiliser secrets-manager CreateSecret
- Utiliser secrets-manager DeleteSecret
- Utiliser secrets-manager GetSecretValue
- Utiliser redshift CreateNamespace
- Utiliser redshift CreateWorkgroup
- Utiliser redshift DeleteNamespace
- Utiliser iam CreateRole

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Amazon Redshift Serverless Tutorial Script with Secrets Manager (No jq
 dependency)
# This script creates a Redshift Serverless environment, loads sample data, and
 runs queries
# Uses AWS Secrets Manager for secure password management without requiring jq

# Set up logging
LOG_FILE="redshift-serverless-tutorial-v4.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Redshift Serverless tutorial script at $(date)"
```

```
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to check for errors in command output
check_error() {
  local output=$1
  local cmd=$2

  if echo "$output" | grep -i "error\|exception\|fail" > /dev/null; then
    echo "ERROR: Command failed: $cmd"
    echo "Output: $output"
    cleanup_resources
    exit 1
  fi
}

# Function to generate a secure password that meets Redshift requirements
generate_secure_password() {
  # Redshift password requirements:
  # - 8-64 characters
  # - At least one uppercase letter
  # - At least one lowercase letter
  # - At least one decimal digit
  # - Can contain printable ASCII characters except /, ", ', \, @, space

  local password=""
  local valid=false
  local attempts=0
  local max_attempts=10

  while [[ "$valid" == false && $attempts -lt $max_attempts ]]; do
    # Generate base password with safe characters
    local base=$(openssl rand -base64 12 | tr -d '/+=' | head -c 12)

    # Ensure we have at least one of each required character type
    local upper=$(echo "ABCDEFGHIJKLMNOPQRSTUVWXYZ" | fold -w1 | shuf -n1)
    local lower=$(echo "abcdefghijklmnopqrstuvwxyz" | fold -w1 | shuf -n1)
    local digit=$(echo "0123456789" | fold -w1 | shuf -n1)
    local special=$(echo "!#$%&*()_+-=[]{}|;:,.<>?" | fold -w1 | shuf -n1)

    # Combine and shuffle
    password="${base}${upper}${lower}${digit}${special}"
    password=$(echo "$password" | fold -w1 | shuf | tr -d '\n')

    # Validate password meets requirements
```

```
    if [[ ${#password} -ge 8 && ${#password} -le 64 ]] && \
       [[ "$password" =~ [A-Z] ]] && \
       [[ "$password" =~ [a-z] ]] && \
       [[ "$password" =~ [0-9] ]] && \
       [[ ! "$password" =~ [/\"\'\\@[:space:]] ]]; then
      valid=true
    fi

    ((attempts++))
  done

  if [[ "$valid" == false ]]; then
    echo "ERROR: Failed to generate valid password after $max_attempts attempts"
    exit 1
  fi

  echo "$password"
}

# Function to create secret in AWS Secrets Manager
create_secret() {
  local secret_name=$1
  local username=$2
  local password=$3
  local description=$4

  echo "Creating secret in AWS Secrets Manager: $secret_name"

  # Create the secret using AWS CLI without jq
  local secret_output=$(aws secretsmanager create-secret \
    --name "$secret_name" \
    --description "$description" \
    --secret-string "{\"username\":\"$username\",\"password\":\"$password\"}" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless
 2>&1)

  if echo "$secret_output" | grep -i "error\|exception\|fail" > /dev/null; then
    echo "ERROR: Failed to create secret: $secret_output"
    return 1
  fi

  echo "Secret created successfully: $secret_name"
  return 0
}
```

```
# Function to retrieve password from AWS Secrets Manager
get_password_from_secret() {
  local secret_name=$1

  # Get the secret value and extract password using sed/grep instead of jq
  local secret_value=$(aws secretsmanager get-secret-value \
    --secret-id "$secret_name" \
    --query 'SecretString' \
    --output text 2>/dev/null)

  if [[ $? -eq 0 ]]; then
    # Extract password from JSON using sed
    echo "$secret_value" | sed -n 's/.*"password":"\([^"]*\)".*/\1/p'
  else
    echo ""
  fi
}

# Function to wait for a resource to be available
wait_for_resource() {
  local resource_type=$1
  local resource_name=$2
  local max_attempts=$3
  local wait_seconds=$4
  local check_cmd=$5

  echo "Waiting for $resource_type $resource_name to be available..."

  for ((i=1; i<=$max_attempts; i++)); do
    local output=$($check_cmd 2>/dev/null)
    local status=$(echo "$output" | grep -o '"Status": "[^"]*' | cut -d'"' -f4 ||
 echo "")

    if [[ "$status" == "AVAILABLE" ]]; then
      echo "$resource_type $resource_name is now available"
      return 0
    fi

    echo "Attempt $i/$max_attempts: $resource_type $resource_name status:
 $status. Waiting $wait_seconds seconds..."
    sleep $wait_seconds
  done
```

```
  echo "ERROR: Timed out waiting for $resource_type $resource_name to be
 available"
  return 1
}

# Function to wait for a resource to be deleted
wait_for_resource_deletion() {
  local resource_type=$1
  local resource_name=$2
  local max_attempts=$3
  local wait_seconds=$4
  local check_cmd=$5

  echo "Waiting for $resource_type $resource_name to be deleted..."

  for ((i=1; i<=$max_attempts; i++)); do
    local output=$($check_cmd 2>&1)

    if echo "$output" | grep -i "not found\|does not exist" > /dev/null; then
      echo "$resource_type $resource_name has been deleted"
      return 0
    fi

    echo "Attempt $i/$max_attempts: $resource_type $resource_name is still being
 deleted. Waiting $wait_seconds seconds..."
    sleep $wait_seconds
  done

  echo "ERROR: Timed out waiting for $resource_type $resource_name to be deleted"
  return 1
}

# Function to clean up resources
cleanup_resources() {
  echo ""
  echo "========================================="
  echo "CLEANUP CONFIRMATION"
  echo "========================================="
  echo "The following resources were created:"
  echo "- Redshift Serverless Workgroup: $WORKGROUP_NAME"
  echo "- Redshift Serverless Namespace: $NAMESPACE_NAME"
  echo "- IAM Role: $ROLE_NAME"
  echo "- Secrets Manager Secret: $SECRET_NAME"
  echo ""
```

```
  echo "Do you want to clean up all created resources? (y/n): "
  read -r CLEANUP_CHOICE

  if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."

    # Delete the workgroup
    echo "Deleting Redshift Serverless workgroup $WORKGROUP_NAME..."
    WORKGROUP_DELETE_OUTPUT=$(aws redshift-serverless delete-workgroup --
workgroup-name "$WORKGROUP_NAME" 2>&1)
    echo "$WORKGROUP_DELETE_OUTPUT"

    # Wait for workgroup to be deleted before deleting namespace
    wait_for_resource_deletion "workgroup" "$WORKGROUP_NAME" 20 30 "aws redshift-
serverless get-workgroup --workgroup-name $WORKGROUP_NAME"

    # Delete the namespace
    echo "Deleting Redshift Serverless namespace $NAMESPACE_NAME..."
    NAMESPACE_DELETE_OUTPUT=$(aws redshift-serverless delete-namespace --
namespace-name "$NAMESPACE_NAME" 2>&1)
    echo "$NAMESPACE_DELETE_OUTPUT"

    # Wait for namespace to be deleted
    wait_for_resource_deletion "namespace" "$NAMESPACE_NAME" 20 30 "aws redshift-
serverless get-namespace --namespace-name $NAMESPACE_NAME"

    # Delete the IAM role policy
    echo "Deleting IAM role policy..."
    POLICY_DELETE_OUTPUT=$(aws iam delete-role-policy --role-name "$ROLE_NAME" --
policy-name S3Access 2>&1)
    echo "$POLICY_DELETE_OUTPUT"

    # Delete the IAM role
    echo "Deleting IAM role $ROLE_NAME..."
    ROLE_DELETE_OUTPUT=$(aws iam delete-role --role-name "$ROLE_NAME" 2>&1)
    echo "$ROLE_DELETE_OUTPUT"

    # Delete the secret
    echo "Deleting Secrets Manager secret $SECRET_NAME..."
    SECRET_DELETE_OUTPUT=$(aws secretsmanager delete-secret --secret-id
 "$SECRET_NAME" --force-delete-without-recovery 2>&1)
    echo "$SECRET_DELETE_OUTPUT"

    echo "Cleanup completed."
```

```
  else
    echo "Cleanup skipped. Resources will remain in your AWS account."
  fi
}

# Check if required tools are available
if ! command -v openssl &> /dev/null; then
    echo "ERROR: openssl is required but not installed. Please install openssl to
 continue."
    exit 1
fi

# Generate unique names for resources
RANDOM_SUFFIX=$(cat /dev/urandom | tr -dc 'a-z0-9' | head -c 6)
NAMESPACE_NAME="rs-namespace-${RANDOM_SUFFIX}"
WORKGROUP_NAME="rs-workgroup-${RANDOM_SUFFIX}"
ROLE_NAME="RedshiftServerlessS3Role-${RANDOM_SUFFIX}"
SECRET_NAME="redshift-serverless-admin-${RANDOM_SUFFIX}"
DB_NAME="dev"
ADMIN_USERNAME="admin"

# Generate secure password
echo "Generating secure password..."
ADMIN_PASSWORD=$(generate_secure_password)

# Create secret in AWS Secrets Manager
create_secret "$SECRET_NAME" "$ADMIN_USERNAME" "$ADMIN_PASSWORD" "Admin
 credentials for Redshift Serverless namespace $NAMESPACE_NAME"
if [[ $? -ne 0 ]]; then
  echo "ERROR: Failed to create secret in AWS Secrets Manager"
  exit 1
fi

# Track created resources
CREATED_RESOURCES=()

echo "Using the following resource names:"
echo "- Namespace: $NAMESPACE_NAME"
echo "- Workgroup: $WORKGROUP_NAME"
echo "- IAM Role: $ROLE_NAME"
echo "- Secret: $SECRET_NAME"
echo "- Database: $DB_NAME"
echo "- Admin Username: $ADMIN_USERNAME"
echo "- Admin Password: [STORED IN SECRETS MANAGER]"
```

```
# Step 1: Create IAM role for S3 access
echo "Creating IAM role for Redshift Serverless S3 access..."

# Create trust policy document
cat > redshift-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift-serverless.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create S3 access policy document
cat > redshift-s3-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::redshift-downloads",
        "arn:aws:s3:::redshift-downloads/*"
      ]
    }
  ]
}
EOF

# Create IAM role
echo "Creating IAM role $ROLE_NAME..."
ROLE_OUTPUT=$(aws iam create-role --role-name "$ROLE_NAME" --assume-role-policy-
document file://redshift-trust-policy.json 2>&1)
```

```
echo "$ROLE_OUTPUT"
check_error "$ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless
CREATED_RESOURCES+=("IAM Role: $ROLE_NAME")

# Attach S3 policy to the role
echo "Attaching S3 access policy to role $ROLE_NAME..."
POLICY_OUTPUT=$(aws iam put-role-policy --role-name "$ROLE_NAME" --policy-name
 S3Access --policy-document file://redshift-s3-policy.json 2>&1)
echo "$POLICY_OUTPUT"
check_error "$POLICY_OUTPUT" "aws iam put-role-policy"

# Get the role ARN
ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn' --output
 text)
echo "Role ARN: $ROLE_ARN"

# Step 2: Create a namespace
echo "Creating Redshift Serverless namespace $NAMESPACE_NAME..."
NAMESPACE_OUTPUT=$(aws redshift-serverless create-namespace \
  --namespace-name "$NAMESPACE_NAME" \
  --admin-username "$ADMIN_USERNAME" \
  --admin-user-password "$ADMIN_PASSWORD" \
  --db-name "$DB_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless 2>&1)
echo "$NAMESPACE_OUTPUT"
check_error "$NAMESPACE_OUTPUT" "aws redshift-serverless create-namespace"
CREATED_RESOURCES+=("Redshift Serverless Namespace: $NAMESPACE_NAME")

# Wait for namespace to be available
wait_for_resource "namespace" "$NAMESPACE_NAME" 10 30 "aws redshift-serverless
 get-namespace --namespace-name $NAMESPACE_NAME"

# Associate IAM role with namespace
echo "Associating IAM role with namespace..."
UPDATE_NAMESPACE_OUTPUT=$(aws redshift-serverless update-namespace \
  --namespace-name "$NAMESPACE_NAME" \
  --iam-roles "$ROLE_ARN" 2>&1)
echo "$UPDATE_NAMESPACE_OUTPUT"
check_error "$UPDATE_NAMESPACE_OUTPUT" "aws redshift-serverless update-namespace"

# Step 3: Create a workgroup
echo "Creating Redshift Serverless workgroup $WORKGROUP_NAME..."
```

```
WORKGROUP_OUTPUT=$(aws redshift-serverless create-workgroup \
  --workgroup-name "$WORKGROUP_NAME" \
  --namespace-name "$NAMESPACE_NAME" \
  --base-capacity 8 \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless 2>&1)
echo "$WORKGROUP_OUTPUT"
check_error "$WORKGROUP_OUTPUT" "aws redshift-serverless create-workgroup"
CREATED_RESOURCES+=("Redshift Serverless Workgroup: $WORKGROUP_NAME")

# Wait for workgroup to be available
wait_for_resource "workgroup" "$WORKGROUP_NAME" 20 30 "aws redshift-serverless
 get-workgroup --workgroup-name $WORKGROUP_NAME"

# Get workgroup endpoint
WORKGROUP_ENDPOINT=$(aws redshift-serverless get-workgroup \
  --workgroup-name "$WORKGROUP_NAME" \
  --query 'workgroup.endpoint.address' \
  --output text)
echo "Workgroup endpoint: $WORKGROUP_ENDPOINT"

# Wait additional time for the endpoint to be fully operational
echo "Waiting for endpoint to be fully operational..."
sleep 60

# Step 4: Create tables for sample data
echo "Creating tables for sample data..."

# Create users table
echo "Creating users table..."
USERS_TABLE_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "CREATE TABLE users(
    userid INTEGER NOT NULL DISTKEY SORTKEY,
    username CHAR(8),
    firstname VARCHAR(30),
    lastname VARCHAR(30),
    city VARCHAR(30),
    state CHAR(2),
    email VARCHAR(100),
    phone CHAR(14),
    likesports BOOLEAN,
    liketheatre BOOLEAN,
    likeconcerts BOOLEAN,
```

```
    likejazz BOOLEAN,
    likeclassical BOOLEAN,
    likeopera BOOLEAN,
    likerock BOOLEAN,
    likevegas BOOLEAN,
    likebroadway BOOLEAN,
    likemusicals BOOLEAN
  );" 2>&1)
echo "$USERS_TABLE_OUTPUT"
check_error "$USERS_TABLE_OUTPUT" "aws redshift-data execute-statement (users
 table)"
USERS_QUERY_ID=$(echo "$USERS_TABLE_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"'
 -f4)

# Wait for query to complete
echo "Waiting for users table creation to complete..."
sleep 5

# Create event table
echo "Creating event table..."
EVENT_TABLE_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "CREATE TABLE event(
    eventid INTEGER NOT NULL DISTKEY,
    venueid SMALLINT NOT NULL,
    catid SMALLINT NOT NULL,
    dateid SMALLINT NOT NULL SORTKEY,
    eventname VARCHAR(200),
    starttime TIMESTAMP
  );" 2>&1)
echo "$EVENT_TABLE_OUTPUT"
check_error "$EVENT_TABLE_OUTPUT" "aws redshift-data execute-statement (event
 table)"
EVENT_QUERY_ID=$(echo "$EVENT_TABLE_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"'
 -f4)

# Wait for query to complete
echo "Waiting for event table creation to complete..."
sleep 5

# Create sales table
echo "Creating sales table..."
SALES_TABLE_OUTPUT=$(aws redshift-data execute-statement \
```

```
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "CREATE TABLE sales(
    salesid INTEGER NOT NULL,
    listid INTEGER NOT NULL DISTKEY,
    sellerid INTEGER NOT NULL,
    buyerid INTEGER NOT NULL,
    eventid INTEGER NOT NULL,
    dateid SMALLINT NOT NULL SORTKEY,
    qtysold SMALLINT NOT NULL,
    pricepaid DECIMAL(8,2),
    commission DECIMAL(8,2),
    saletime TIMESTAMP
  );" 2>&1)
echo "$SALES_TABLE_OUTPUT"
check_error "$SALES_TABLE_OUTPUT" "aws redshift-data execute-statement (sales
 table)"
SALES_QUERY_ID=$(echo "$SALES_TABLE_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"'
 -f4)

# Wait for tables to be created
echo "Waiting for tables to be created..."
sleep 10

# Step 5: Load sample data from Amazon S3
echo "Loading sample data from Amazon S3..."

# Load data into users table
echo "Loading data into users table..."
USERS_LOAD_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "COPY users
    FROM 's3://redshift-downloads/tickit/allusers_pipe.txt'
    DELIMITER '|'
    TIMEFORMAT 'YYYY-MM-DD HH:MI:SS'
    IGNOREHEADER 1
    IAM_ROLE '$ROLE_ARN';" 2>&1)
echo "$USERS_LOAD_OUTPUT"
check_error "$USERS_LOAD_OUTPUT" "aws redshift-data execute-statement (load
 users)"
USERS_LOAD_QUERY_ID=$(echo "$USERS_LOAD_OUTPUT" | grep -o '"Id": "[^"]*' | cut -
d'"' -f4)
```

```
# Wait for data loading to complete
echo "Waiting for users data loading to complete..."
sleep 10

# Load data into event table
echo "Loading data into event table..."
EVENT_LOAD_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "COPY event
    FROM 's3://redshift-downloads/tickit/allevents_pipe.txt'
    DELIMITER '|'
    TIMEFORMAT 'YYYY-MM-DD HH:MI:SS'
    IGNOREHEADER 1
    IAM_ROLE '$ROLE_ARN';" 2>&1)
echo "$EVENT_LOAD_OUTPUT"
check_error "$EVENT_LOAD_OUTPUT" "aws redshift-data execute-statement (load
 event)"
EVENT_LOAD_QUERY_ID=$(echo "$EVENT_LOAD_OUTPUT" | grep -o '"Id": "[^"]*' | cut -
d'"' -f4)

# Wait for data loading to complete
echo "Waiting for event data loading to complete..."
sleep 10

# Load data into sales table
echo "Loading data into sales table..."
SALES_LOAD_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "COPY sales
    FROM 's3://redshift-downloads/tickit/sales_tab.txt'
    DELIMITER '\t'
    TIMEFORMAT 'MM/DD/YYYY HH:MI:SS'
    IGNOREHEADER 1
    IAM_ROLE '$ROLE_ARN';" 2>&1)
echo "$SALES_LOAD_OUTPUT"
check_error "$SALES_LOAD_OUTPUT" "aws redshift-data execute-statement (load
 sales)"
SALES_LOAD_QUERY_ID=$(echo "$SALES_LOAD_OUTPUT" | grep -o '"Id": "[^"]*' | cut -
d'"' -f4)

# Wait for data loading to complete
echo "Waiting for sales data loading to complete..."
```

```
sleep 30

# Step 6: Run sample queries
echo "Running sample queries..."

# Query 1: Find top 10 buyers by quantity
echo "Running query: Find top 10 buyers by quantity..."
QUERY1_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "SELECT firstname, lastname, total_quantity
    FROM (SELECT buyerid, sum(qtysold) total_quantity
          FROM sales
          GROUP BY buyerid
          ORDER BY total_quantity desc limit 10) Q, users
    WHERE Q.buyerid = userid
    ORDER BY Q.total_quantity desc;" 2>&1)
echo "$QUERY1_OUTPUT"
check_error "$QUERY1_OUTPUT" "aws redshift-data execute-statement (query 1)"
QUERY1_ID=$(echo "$QUERY1_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"' -f4)

# Wait for query to complete
echo "Waiting for query 1 to complete..."
sleep 10

# Get query 1 results
echo "Getting results for query 1..."
QUERY1_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY1_ID"
 2>&1)
echo "$QUERY1_STATUS_OUTPUT"
check_error "$QUERY1_STATUS_OUTPUT" "aws redshift-data describe-statement (query
 1)"

QUERY1_STATUS=$(echo "$QUERY1_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut -
d'"' -f4)
if [ "$QUERY1_STATUS" == "FINISHED" ]; then
  QUERY1_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY1_ID" 2>&1)
  echo "Query 1 Results:"
  echo "$QUERY1_RESULTS"
else
  echo "Query 1 is not yet complete. Status: $QUERY1_STATUS"
  echo "Waiting additional time for query to complete..."
  sleep 20
```

```
  # Check again
  QUERY1_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY1_ID"
 2>&1)
  QUERY1_STATUS=$(echo "$QUERY1_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut
 -d'"' -f4)

  if [ "$QUERY1_STATUS" == "FINISHED" ]; then
    QUERY1_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY1_ID"
 2>&1)
    echo "Query 1 Results:"
    echo "$QUERY1_RESULTS"
  else
    echo "Query 1 is still not complete. Status: $QUERY1_STATUS"
  fi
fi

# Query 2: Find events in the 99.9 percentile in terms of all time total sales
echo "Running query: Find events in the 99.9 percentile in terms of all time
 total sales..."
QUERY2_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "SELECT eventname, total_price
    FROM (SELECT eventid, total_price, ntile(1000) over(order by total_price
 desc) as percentile
          FROM (SELECT eventid, sum(pricepaid) total_price
                FROM sales
                GROUP BY eventid)) Q, event E
    WHERE Q.eventid = E.eventid
    AND percentile = 1
    ORDER BY total_price desc;" 2>&1)
echo "$QUERY2_OUTPUT"
check_error "$QUERY2_OUTPUT" "aws redshift-data execute-statement (query 2)"
QUERY2_ID=$(echo "$QUERY2_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"' -f4)

# Wait for query to complete
echo "Waiting for query 2 to complete..."
sleep 10

# Get query 2 results
echo "Getting results for query 2..."
QUERY2_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY2_ID"
 2>&1)
echo "$QUERY2_STATUS_OUTPUT"
```

```
check_error "$QUERY2_STATUS_OUTPUT" "aws redshift-data describe-statement (query
 2)"

QUERY2_STATUS=$(echo "$QUERY2_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut -
d'"' -f4)
if [ "$QUERY2_STATUS" == "FINISHED" ]; then
  QUERY2_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY2_ID" 2>&1)
  echo "Query 2 Results:"
  echo "$QUERY2_RESULTS"
else
  echo "Query 2 is not yet complete. Status: $QUERY2_STATUS"
  echo "Waiting additional time for query to complete..."
  sleep 20

  # Check again
  QUERY2_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY2_ID"
 2>&1)
  QUERY2_STATUS=$(echo "$QUERY2_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut
 -d'"' -f4)

  if [ "$QUERY2_STATUS" == "FINISHED" ]; then
    QUERY2_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY2_ID"
 2>&1)
    echo "Query 2 Results:"
    echo "$QUERY2_RESULTS"
  else
    echo "Query 2 is still not complete. Status: $QUERY2_STATUS"
  fi
fi

# Summary
echo ""
echo "=========================================="
echo "TUTORIAL SUMMARY"
echo "=========================================="
echo "You have successfully:"
echo "1. Created a Redshift Serverless namespace and workgroup"
echo "2. Created an IAM role with S3 access permissions"
echo "3. Stored admin credentials securely in AWS Secrets Manager"
echo "4. Created tables for sample data"
echo "5. Loaded sample data from Amazon S3"
echo "6. Run sample queries on the data"
echo ""
echo "Redshift Serverless Resources:"
```

```
echo "- Namespace: $NAMESPACE_NAME"
echo "- Workgroup: $WORKGROUP_NAME"
echo "- Database: $DB_NAME"
echo "- Endpoint: $WORKGROUP_ENDPOINT"
echo "- Credentials Secret: $SECRET_NAME"
echo ""
echo "To connect to your Redshift Serverless database using SQL tools:"
echo "- Host: $WORKGROUP_ENDPOINT"
echo "- Database: $DB_NAME"
echo "- Username: $ADMIN_USERNAME"
echo "- Password: Retrieve from AWS Secrets Manager secret '$SECRET_NAME'"
echo ""
echo "To retrieve the password from Secrets Manager (without jq):"
echo "aws secretsmanager get-secret-value --secret-id $SECRET_NAME --query
 'SecretString' --output text | sed -n 's/.*\"password\":\"\([^\"]*\)\".*/\1/p'"
echo ""

# Clean up temporary files
rm -f redshift-trust-policy.json redshift-s3-policy.json

# Clean up resources
cleanup_resources

echo "Tutorial completed at $(date)"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - CreateNamespace
  - CreateRole
  - CreateSecret
  - CreateWorkgroup
  - DeleteNamespace
  - DeleteRole
  - DeleteRolePolicy
  - DeleteSecret
  - DeleteWorkgroup
  - GetNamespace
  - GetRole

- GetSecretValue
- GetWorkgroup
- PutRolePolicy
- UpdateNamespace

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Commencer à gérer la configuration

L'exemple de code suivant illustre comment :

- Créer un compartiment Amazon S3
- Créer une rubrique Amazon SNS
- Création d'un rôle IAM pour Config
- Configurer l'enregistreur de configuration Config
- Configurer le canal de diffusion Config
- Démarrez l'enregistreur de configuration
- Vérifiez la configuration de Config

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# AWS Config Setup Script (v2)
# This script sets up AWS Config with the AWS CLI
```

```
# Error handling
set -e
LOGFILE="aws-config-setup-v2.log"
touch $LOGFILE
exec > >(tee -a $LOGFILE)
exec 2>&1

# Function to handle errors
handle_error() {
    echo "ERROR: An error occurred at line $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set trap for error handling
trap 'handle_error $LINENO' ERR

# Function to generate random identifier
generate_random_id() {
    echo $(openssl rand -hex 6)
}

# Function to check if command was successful
check_command() {
    if echo "$1" | grep -i "error" > /dev/null; then
        echo "ERROR: $1"
        return 1
    fi
    return 0
}

# Function to clean up resources
cleanup_resources() {
    if [ -n "$CONFIG_RECORDER_NAME" ]; then
        echo "Stopping configuration recorder..."
        aws configservice stop-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    # Check if we created a new delivery channel before trying to delete it
    if [ -n "$DELIVERY_CHANNEL_NAME" ] && [ "$CREATED_NEW_DELIVERY_CHANNEL" =
 "true" ]; then
        echo "Deleting delivery channel..."
```

```
        aws configservice delete-delivery-channel --delivery-channel-name
 "$DELIVERY_CHANNEL_NAME" 2>/dev/null || true
    fi

    if [ -n "$CONFIG_RECORDER_NAME" ] && [ "$CREATED_NEW_CONFIG_RECORDER" =
 "true" ]; then
        echo "Deleting configuration recorder..."
        aws configservice delete-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    if [ -n "$ROLE_NAME" ]; then
        if [ -n "$POLICY_NAME" ]; then
            echo "Detaching custom policy from role..."
            aws iam delete-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" 2>/dev/null || true
        fi

        if [ -n "$MANAGED_POLICY_ARN" ]; then
            echo "Detaching managed policy from role..."
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN" 2>/dev/null || true
        fi

        echo "Deleting IAM role..."
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi

    if [ -n "$SNS_TOPIC_ARN" ]; then
        echo "Deleting SNS topic..."
        aws sns delete-topic --topic-arn "$SNS_TOPIC_ARN" 2>/dev/null || true
    fi

    if [ -n "$S3_BUCKET_NAME" ]; then
        echo "Emptying S3 bucket..."
        aws s3 rm "s3://$S3_BUCKET_NAME" --recursive 2>/dev/null || true

        echo "Deleting S3 bucket..."
        if [ "$BUCKET_IS_SHARED" = "false" ]; then
            aws s3api delete-bucket --bucket "$S3_BUCKET_NAME" 2>/dev/null ||
 true
        fi
    fi
}
```

```
# Function to display created resources
display_resources() {
    echo ""
    echo "========================================="
    echo "CREATED RESOURCES"
    echo "========================================="
    echo "S3 Bucket: $S3_BUCKET_NAME"
    echo "SNS Topic ARN: $SNS_TOPIC_ARN"
    echo "IAM Role: $ROLE_NAME"
    if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (newly created)"
    else
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (existing)"
    fi
    if [ "$CREATED_NEW_DELIVERY_CHANNEL" = "true" ]; then
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (newly created)"
    else
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (existing)"
    fi
    echo "========================================="
}

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
if [ -z "$ACCOUNT_ID" ]; then
    echo "ERROR: Failed to get AWS account ID"
    exit 1
fi
echo "AWS Account ID: $ACCOUNT_ID"

# Generate random identifier for resources
RANDOM_ID=$(generate_random_id)
echo "Generated random identifier: $RANDOM_ID"

# Step 1: Create an S3 bucket
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
```

```
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="configservice-${RANDOM_ID}"
    echo "Creating S3 bucket: $S3_BUCKET_NAME"
fi

# Get the current region
AWS_REGION=$(aws configure get region)
if [ -z "$AWS_REGION" ]; then
    AWS_REGION="us-east-1"  # Default to us-east-1 if no region is configured
fi
echo "Using AWS Region: $AWS_REGION"

# Create bucket with appropriate command based on region
if [ "$BUCKET_IS_SHARED" = "false" ]; then
    if [ "$AWS_REGION" = "us-east-1" ]; then
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME")
    else
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" --
create-bucket-configuration LocationConstraint="$AWS_REGION")
    fi
    check_command "$BUCKET_RESULT"
    echo "S3 bucket created: $S3_BUCKET_NAME"

    aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-config-gs}]'
    echo "Tags applied to S3 bucket"
else
    echo "Using shared bucket: $S3_BUCKET_NAME (skipping creation)"
fi

# Block public access for the bucket
aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
echo "Public access blocked for bucket"

# Step 2: Create an SNS topic
TOPIC_NAME="config-topic-${RANDOM_ID}"
echo "Creating SNS topic: $TOPIC_NAME"
```

```
SNS_RESULT=$(aws sns create-topic --name "$TOPIC_NAME" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs)
check_command "$SNS_RESULT"
SNS_TOPIC_ARN=$(echo "$SNS_RESULT" | grep -o 'arn:aws:sns:[^"]*')
echo "SNS topic created: $SNS_TOPIC_ARN"

# Step 3: Create an IAM role for AWS Config
ROLE_NAME="config-role-${RANDOM_ID}"
POLICY_NAME="config-delivery-permissions"
MANAGED_POLICY_ARN="arn:aws:iam::aws:policy/service-role/AWS_ConfigRole"

echo "Creating trust policy document..."
cat > config-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "config.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role --role-name "$ROLE_NAME" --assume-role-policy-
document file://config-trust-policy.json)
check_command "$ROLE_RESULT"
ROLE_ARN=$(echo "$ROLE_RESULT" | grep -o 'arn:aws:iam::[^"]*' | head -1)
echo "IAM role created: $ROLE_ARN"

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-config-gs
echo "Tags applied to IAM role"

echo "Attaching AWS managed policy to role..."
ATTACH_RESULT=$(aws iam attach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN")
check_command "$ATTACH_RESULT"
echo "AWS managed policy attached"
```

```
echo "Creating custom policy document for S3 and SNS access..."
cat > config-delivery-permissions.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}/AWSLogs/${ACCOUNT_ID}/*",
      "Condition": {
        "StringLike": {
          "s3:x-amz-acl": "bucket-owner-full-control"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketAcl"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}"
    },
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": "${SNS_TOPIC_ARN}"
    }
  ]
}
EOF

echo "Attaching custom policy to role..."
POLICY_RESULT=$(aws iam put-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" --policy-document file://config-delivery-permissions.json)
check_command "$POLICY_RESULT"
echo "Custom policy attached"

# Wait for IAM role to propagate
echo "Waiting for IAM role to propagate (15 seconds)..."
sleep 15
```

```
# Step 4: Check if configuration recorder already exists
CONFIG_RECORDER_NAME="default"
CREATED_NEW_CONFIG_RECORDER="false"

echo "Checking for existing configuration recorder..."
EXISTING_RECORDERS=$(aws configservice describe-configuration-recorders 2>/dev/
null || echo "")
if echo "$EXISTING_RECORDERS" | grep -q "name"; then
    echo "Configuration recorder already exists. Will update it."
    # Get the name of the existing recorder
    CONFIG_RECORDER_NAME=$(echo "$EXISTING_RECORDERS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
    echo "Using existing configuration recorder: $CONFIG_RECORDER_NAME"
else
    echo "No existing configuration recorder found. Will create a new one."
    CREATED_NEW_CONFIG_RECORDER="true"
fi

echo "Creating configuration recorder configuration..."
cat > configurationRecorder.json << EOF
{
  "name": "${CONFIG_RECORDER_NAME}",
  "roleARN": "${ROLE_ARN}",
  "recordingMode": {
    "recordingFrequency": "CONTINUOUS"
  }
}
EOF

echo "Creating recording group configuration..."
cat > recordingGroup.json << EOF
{
  "allSupported": true,
  "includeGlobalResourceTypes": true
}
EOF

echo "Setting up configuration recorder..."
RECORDER_RESULT=$(aws configservice put-configuration-recorder --configuration-
recorder file://configurationRecorder.json --recording-group file://
recordingGroup.json)
check_command "$RECORDER_RESULT"
echo "Configuration recorder set up"
```

```
if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:config-recorder/${CONFIG_RECORDER_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to configuration recorder"
fi

# Step 5: Check if delivery channel already exists
DELIVERY_CHANNEL_NAME="default"
CREATED_NEW_DELIVERY_CHANNEL="false"

echo "Checking for existing delivery channel..."
EXISTING_CHANNELS=$(aws configservice describe-delivery-channels 2>/dev/null ||
 echo "")
if echo "$EXISTING_CHANNELS" | grep -q "name"; then
    echo "Delivery channel already exists."
    # Get the name of the existing channel
    DELIVERY_CHANNEL_NAME=$(echo "$EXISTING_CHANNELS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
    echo "Using existing delivery channel: $DELIVERY_CHANNEL_NAME"

    # Update the existing delivery channel
    echo "Creating delivery channel configuration for update..."
    cat > deliveryChannel.json << EOF
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Updating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel updated"
else
    echo "No existing delivery channel found. Will create a new one."
    CREATED_NEW_DELIVERY_CHANNEL="true"
```

```
    echo "Creating delivery channel configuration..."
    cat > deliveryChannel.json << EOF
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Creating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel created"

    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:delivery-channel/${DELIVERY_CHANNEL_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to delivery channel"
fi

# Step 6: Start the configuration recorder
echo "Checking configuration recorder status..."
RECORDER_STATUS=$(aws configservice describe-configuration-recorder-status 2>/
dev/null || echo "")
if echo "$RECORDER_STATUS" | grep -q '"recording": true'; then
    echo "Configuration recorder is already running."
else
    echo "Starting configuration recorder..."
    START_RESULT=$(aws configservice start-configuration-recorder --
configuration-recorder-name "$CONFIG_RECORDER_NAME")
    check_command "$START_RESULT"
    echo "Configuration recorder started"
fi

# Step 7: Verify the AWS Config setup
echo "Verifying delivery channel..."
VERIFY_CHANNEL=$(aws configservice describe-delivery-channels)
check_command "$VERIFY_CHANNEL"
echo "$VERIFY_CHANNEL"
```

```
echo "Verifying configuration recorder..."
VERIFY_RECORDER=$(aws configservice describe-configuration-recorders)
check_command "$VERIFY_RECORDER"
echo "$VERIFY_RECORDER"

echo "Verifying configuration recorder status..."
VERIFY_STATUS=$(aws configservice describe-configuration-recorder-status)
check_command "$VERIFY_STATUS"
echo "$VERIFY_STATUS"

# Display created resources
display_resources

# Ask if user wants to clean up resources
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE='y'

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..."
    cleanup_resources
    echo "Cleanup completed."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
fi

echo "Script completed successfully!"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreateBucket
  - CreateRole
  - CreateTopic
  - DeleteBucket
  - DeleteConfigurationRecorder

- DeleteDeliveryChannel
- DeleteRole
- DeleteRolePolicy
- DeleteTopic
- DescribeConfigurationRecorderStatus
- DescribeConfigurationRecorders
- DescribeDeliveryChannels
- DetachRolePolicy
- GetCallerIdentity
- PutConfigurationRecorder
- PutDeliveryChannel
- PutPublicAccessBlock
- PutRolePolicy
- Rm
- StartConfigurationRecorder
- StopConfigurationRecorder

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Débuter avec la protection des appareils liés à l'Internet des objets

L'exemple de code suivant illustre comment :

- Création des rôles IAM requis
- Activez les vérifications d'audit d'IoT Device Defender
- Exécuter un On-Demand audit
- Création d'une action d'atténuation
- Appliquer des mesures d'atténuation aux résultats
- Configurer les notifications SNS (facultatif)
- Activer la journalisation de l'IoT

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# AWS IoT Device Defender Getting Started Script
# This script demonstrates how to use AWS IoT Device Defender to enable audit
 checks,
# view audit results, create mitigation actions, and apply them to findings.

set -euo pipefail

# Set up logging
LOG_FILE="iot-device-defender-script-$(date +%Y%m%d%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "================================================="
echo "AWS IoT Device Defender Getting Started Script"
echo "================================================="
echo "Starting script execution at $(date)"
echo ""

# Function to check for errors in command output
check_error() {
    if echo "$1" | grep -iE "An error occurred|Exception|Failed|usage: aws" > /
dev/null; then
        echo "ERROR: Command failed with the following output:"
        echo "$1"
        return 1
    fi
    return 0
}

# Function to safely extract JSON values using jq
extract_json_value() {
```

```
    local json="$1"
    local key="$2"
    echo "$json" | jq -r ".${key} // empty" 2>/dev/null || echo ""
}

# Function to validate JSON
validate_json() {
    local json="$1"
    echo "$json" | jq empty 2>/dev/null
}

# Function to check AWS CLI availability
check_aws_cli() {
    if ! command -v aws &> /dev/null; then
        echo "ERROR: AWS CLI is not installed or not in PATH"
        return 1
    fi
    if ! command -v jq &> /dev/null; then
        echo "ERROR: jq is not installed or not in PATH"
        return 1
    fi
    return 0
}

# Function to get AWS account ID
get_account_id() {
    local account_id
    account_id=$(aws sts get-caller-identity --query 'Account' --output text 2>/
dev/null) || true
    if [ -z "$account_id" ]; then
        echo "ERROR: Could not retrieve AWS account ID"
        return 1
    fi
    echo "$account_id"
    return 0
}

# Function to create IAM roles with retry logic
create_iam_role() {
    local ROLE_NAME=$1
    local TRUST_POLICY=$2
    local MANAGED_POLICY=$3
    local RETRY_COUNT=0
    local MAX_RETRIES=3
```

```
    echo "Creating IAM role: $ROLE_NAME"

    # Validate trust policy JSON
    if ! validate_json "$TRUST_POLICY"; then
        echo "ERROR: Invalid trust policy JSON for role $ROLE_NAME"
        return 1
    fi

    # Check if role already exists
    if aws iam get-role --role-name "$ROLE_NAME" >/dev/null 2>&1; then
        echo "Role $ROLE_NAME already exists, skipping creation"
        ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn'
 --output text 2>/dev/null) || true
        if [ -z "$ROLE_ARN" ]; then
            echo "ERROR: Could not retrieve ARN for existing role $ROLE_NAME"
            return 1
        fi
        echo "Role ARN: $ROLE_ARN"
        return 0
    fi

    # Create the role with trust policy and retry logic
    while [ $RETRY_COUNT -lt $MAX_RETRIES ]; do
        ROLE_RESULT=$(aws iam create-role \
            --role-name "$ROLE_NAME" \
            --assume-role-policy-document "$TRUST_POLICY" 2>&1) || true

        if check_error "$ROLE_RESULT"; then
            break
        fi

        RETRY_COUNT=$((RETRY_COUNT + 1))
        if [ $RETRY_COUNT -lt $MAX_RETRIES ]; then
            echo "Retrying role creation (attempt $((RETRY_COUNT + 1))/
$MAX_RETRIES)..."
            sleep $((RETRY_COUNT * 2))
        fi
    done

    if ! check_error "$ROLE_RESULT"; then
        echo "Failed to create role $ROLE_NAME after $MAX_RETRIES attempts"
        return 1
    fi
```

```
    aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-device-defender-gs 2>&1 || true

    # For IoT logging role, create an inline policy instead of using a managed
 policy
    if [[ "$ROLE_NAME" == "AWSIoTLoggingRole" ]]; then
        local LOGGING_POLICY
        LOGGING_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:PutMetricFilter",
                "logs:PutRetentionPolicy",
                "logs:GetLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": "arn:aws:logs:*:*:*"
        }
    ]
}
EOF
)

        if ! validate_json "$LOGGING_POLICY"; then
            echo "ERROR: Invalid logging policy JSON"
            return 1
        fi

        POLICY_RESULT=$(aws iam put-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-name "${ROLE_NAME}Policy" \
            --policy-document "$LOGGING_POLICY" 2>&1) || true

        if ! check_error "$POLICY_RESULT"; then
            echo "Failed to attach inline policy to role $ROLE_NAME"
            return 1
        fi
```

```
    elif [[ "$ROLE_NAME" == "IoTMitigationActionErrorLoggingRole" ]]; then
        local MITIGATION_POLICY
        MITIGATION_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iot:UpdateCACertificate",
                "iot:UpdateCertificate",
                "iot:SetV2LoggingOptions",
                "iot:SetLoggingOptions",
                "iot:AddThingToThingGroup"
            ],
            "Resource": "arn:aws:iot:*:*:*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "iot.amazonaws.com"
                }
            }
        }
    ]
}
EOF
)

        if ! validate_json "$MITIGATION_POLICY"; then
            echo "ERROR: Invalid mitigation policy JSON"
            return 1
        fi

        POLICY_RESULT=$(aws iam put-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-name "${ROLE_NAME}Policy" \
            --policy-document "$MITIGATION_POLICY" 2>&1) || true

        if ! check_error "$POLICY_RESULT"; then
            echo "Failed to attach inline policy to role $ROLE_NAME"
```

```
            return 1
        fi
    else
        # Attach managed policy to role if provided
        if [ -n "$MANAGED_POLICY" ]; then
            ATTACH_RESULT=$(aws iam attach-role-policy \
                --role-name "$ROLE_NAME" \
                --policy-arn "$MANAGED_POLICY" 2>&1) || true

            if ! check_error "$ATTACH_RESULT"; then
                echo "Failed to attach policy to role $ROLE_NAME"
                return 1
            fi
        fi
    fi

    echo "Role $ROLE_NAME created successfully"

    # Get the role ARN with error handling
    ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn' --
output text 2>/dev/null) || true
    if [ -z "$ROLE_ARN" ]; then
        echo "ERROR: Could not retrieve ARN for newly created role $ROLE_NAME"
        return 1
    fi
    echo "Role ARN: $ROLE_ARN"
    return 0
}

# Array to store created resources for cleanup
declare -a CREATED_RESOURCES

# Validate prerequisites
echo "Validating prerequisites..."
if ! check_aws_cli; then
    echo "ERROR: Prerequisites not met"
    exit 1
fi

ACCOUNT_ID=$(get_account_id) || exit 1
echo "AWS Account ID: $ACCOUNT_ID"
echo ""

# Step 1: Create IAM roles needed for the tutorial
```

```
echo "================================================="
echo "Step 1: Creating required IAM roles"
echo "================================================="

# Create IoT Device Defender Audit role
IOT_DEFENDER_AUDIT_TRUST_POLICY=$(cat <<'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "iot.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
)

if ! create_iam_role "AWSIoTDeviceDefenderAuditRole"
 "$IOT_DEFENDER_AUDIT_TRUST_POLICY" "arn:aws:iam::aws:policy/service-role/
AWSIoTDeviceDefenderAudit"; then
    echo "ERROR: Failed to create audit role"
    exit 1
fi
AUDIT_ROLE_ARN=$ROLE_ARN
CREATED_RESOURCES+=("IAM Role: AWSIoTDeviceDefenderAuditRole")

# Create IoT Logging role
IOT_LOGGING_TRUST_POLICY=$(cat <<'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "iot.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

```
EOF
)

if ! create_iam_role "AWSIoTLoggingRole" "$IOT_LOGGING_TRUST_POLICY" ""; then
    echo "ERROR: Failed to create logging role"
    exit 1
fi
LOGGING_ROLE_ARN=$ROLE_ARN
CREATED_RESOURCES+=("IAM Role: AWSIoTLoggingRole")

# Create IoT Mitigation Action role
IOT_MITIGATION_TRUST_POLICY=$(cat <<'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "iot.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
)

if ! create_iam_role "IoTMitigationActionErrorLoggingRole"
 "$IOT_MITIGATION_TRUST_POLICY" ""; then
    echo "ERROR: Failed to create mitigation role"
    exit 1
fi
MITIGATION_ROLE_ARN=$ROLE_ARN
CREATED_RESOURCES+=("IAM Role: IoTMitigationActionErrorLoggingRole")

# Wait for IAM role propagation
echo "Waiting for IAM role propagation..."
sleep 5

# Step 2: Enable audit checks
echo ""
echo "================================================="
echo "Step 2: Enabling AWS IoT Device Defender audit checks"
echo "================================================="
```

```
# Get current audit configuration
echo "Getting current audit configuration..."
CURRENT_CONFIG=$(aws iot describe-account-audit-configuration --output json 2>&1)
 || true
if validate_json "$CURRENT_CONFIG"; then
    echo "$CURRENT_CONFIG" | jq '.' 2>/dev/null || echo "Could not parse current
 configuration"
fi

# Enable specific audit checks with proper JSON escaping
echo "Enabling audit checks..."
AUDIT_CONFIG='{"LOGGING_DISABLED_CHECK":{"enabled":true}}'

if ! validate_json "$AUDIT_CONFIG"; then
    echo "ERROR: Invalid audit configuration JSON"
    exit 1
fi

UPDATE_RESULT=$(aws iot update-account-audit-configuration \
  --role-arn "$AUDIT_ROLE_ARN" \
  --audit-check-configurations "$AUDIT_CONFIG" 2>&1) || true

if ! check_error "$UPDATE_RESULT"; then
    echo "Failed to update audit configuration"
    exit 1
fi

echo "Audit checks enabled successfully"

# Step 3: Run an on-demand audit
echo ""
echo "================================================="
echo "Step 3: Running an on-demand audit"
echo "================================================="

echo "Starting on-demand audit task..."
AUDIT_TASK_RESULT=$(aws iot start-on-demand-audit-task \
  --target-check-names LOGGING_DISABLED_CHECK --output json 2>&1) || true

if ! check_error "$AUDIT_TASK_RESULT"; then
    echo "Failed to start on-demand audit task"
    exit 1
fi
```

```
TASK_ID=$(extract_json_value "$AUDIT_TASK_RESULT" "taskId")
if [ -z "$TASK_ID" ]; then
    echo "ERROR: Could not extract task ID from response"
    exit 1
fi

echo "Audit task started with ID: $TASK_ID"
CREATED_RESOURCES+=("Audit Task: $TASK_ID")

# Tag the audit task via IoT service
aws iot tag-resource --resource-arn "arn:aws:iot:$(aws configure get region):
${ACCOUNT_ID}:audittask/${TASK_ID}" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-device-defender-gs 2>&1 || true

# Wait for the audit task to complete
echo "Waiting for audit task to complete (this may take a few minutes)..."
TASK_STATUS="IN_PROGRESS"
TIMEOUT=0
MAX_TIMEOUT=600
POLL_INTERVAL=15

while [ "$TASK_STATUS" != "COMPLETED" ]; do
    if [ $TIMEOUT -ge $MAX_TIMEOUT ]; then
        echo "WARNING: Audit task did not complete within ${MAX_TIMEOUT} seconds,
 continuing..."
        break
    fi

    sleep "$POLL_INTERVAL"
    TIMEOUT=$((TIMEOUT + POLL_INTERVAL))

    TASK_DETAILS=$(aws iot describe-audit-task --task-id "$TASK_ID" --output json
 2>&1) || true
    if validate_json "$TASK_DETAILS"; then
        TASK_STATUS=$(extract_json_value "$TASK_DETAILS" "taskStatus")
        echo "Current task status: $TASK_STATUS (elapsed: ${TIMEOUT}s)"

        if [ "$TASK_STATUS" = "FAILED" ]; then
            echo "WARNING: Audit task failed, continuing with script..."
            FAILURE_REASON=$(extract_json_value "$TASK_DETAILS"
 "taskStatistics.failedChecksNotApplicable")
            if [ -n "$FAILURE_REASON" ]; then
                echo "Reason: $FAILURE_REASON"
```

```
            fi
            break
        fi
    else
        echo "WARNING: Could not parse task details, retrying..."
    fi
done

echo "Audit task processing completed"

# Get audit findings (non-blocking)
echo "Getting audit findings..."
FINDINGS=$(aws iot list-audit-findings \
  --task-id "$TASK_ID" --output json 2>&1) || true

if validate_json "$FINDINGS"; then
    FINDING_COUNT=$(echo "$FINDINGS" | jq '.findings | length' 2>/dev/null ||
 echo "0")
    echo "Audit findings count: $FINDING_COUNT"
    if [ "$FINDING_COUNT" -gt 0 ]; then
        echo "Sample finding:"
        echo "$FINDINGS" | jq '.findings[0]' 2>/dev/null || echo "Could not parse
 finding"
    fi
else
    echo "WARNING: Could not parse audit findings response"
    FINDINGS='{"findings":[]}'
fi

# Check if we have any non-compliant findings
FINDING_ID=$(extract_json_value "$FINDINGS" "findings[0].findingId")
if [ -n "$FINDING_ID" ]; then
    echo "Found non-compliant finding with ID: $FINDING_ID"
    HAS_FINDINGS=true
else
    echo "No non-compliant findings detected"
    HAS_FINDINGS=false
fi

# Step 4: Create a mitigation action
echo ""
echo "================================================="
echo "Step 4: Creating a mitigation action"
echo "================================================="
```

```
# Check if mitigation action already exists and delete it
if aws iot describe-mitigation-action --action-name "EnableErrorLoggingAction" >/
dev/null 2>&1; then
    echo "Mitigation action 'EnableErrorLoggingAction' already exists, deleting
 it first..."
    aws iot delete-mitigation-action --action-name "EnableErrorLoggingAction"
 2>&1 || true
    sleep 2
fi

echo "Creating mitigation action to enable AWS IoT logging..."

# Build mitigation action parameters JSON
MITIGATION_PARAMS=$(cat <<EOF
{
  "enableIoTLoggingParams": {
    "roleArnForLogging": "$LOGGING_ROLE_ARN",
    "logLevel": "ERROR"
  }
}
EOF
)

if ! validate_json "$MITIGATION_PARAMS"; then
    echo "ERROR: Invalid mitigation parameters JSON"
    exit 1
fi

MITIGATION_RESULT=$(aws iot create-mitigation-action \
  --action-name "EnableErrorLoggingAction" \
  --role-arn "$MITIGATION_ROLE_ARN" \
  --action-params "$MITIGATION_PARAMS" --output json 2>&1) || true

if ! check_error "$MITIGATION_RESULT"; then
    echo "Failed to create mitigation action"
    exit 1
fi

if validate_json "$MITIGATION_RESULT"; then
    echo "Mitigation action created successfully"
    MITIGATION_ACTION_ARN=$(extract_json_value "$MITIGATION_RESULT" "actionArn")
    if [ -n "$MITIGATION_ACTION_ARN" ]; then
        echo "Mitigation Action ARN: $MITIGATION_ACTION_ARN"
```

```
        aws iot tag-resource --resource-arn "$MITIGATION_ACTION_ARN" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-iot-device-defender-gs 2>&1
 || true
    fi
else
    echo "WARNING: Could not validate mitigation action response, but action may
 have been created"
fi

CREATED_RESOURCES+=("Mitigation Action: EnableErrorLoggingAction")

# Step 5: Apply mitigation action to findings (if any)
if [ "$HAS_FINDINGS" = true ]; then
    echo ""
    echo "=================================================="
    echo "Step 5: Applying mitigation action to findings"
    echo "=================================================="

    MITIGATION_TASK_ID="MitigationTask-$(date +%s)"
    echo "Starting mitigation actions task with ID: $MITIGATION_TASK_ID"

    # Build target JSON
    TARGET_JSON=$(cat <<EOF
{
  "findingIds": ["$FINDING_ID"]
}
EOF
)

    if ! validate_json "$TARGET_JSON"; then
        echo "ERROR: Invalid target JSON"
        exit 1
    fi

    # Build audit check to actions mapping JSON
    AUDIT_CHECK_MAPPING=$(cat <<EOF
{
  "LOGGING_DISABLED_CHECK": ["EnableErrorLoggingAction"]
}
EOF
)

    if ! validate_json "$AUDIT_CHECK_MAPPING"; then
        echo "ERROR: Invalid audit check mapping JSON"
```

```
        exit 1
    fi

    MITIGATION_TASK_RESULT=$(aws iot start-audit-mitigation-actions-task \
      --task-id "$MITIGATION_TASK_ID" \
      --target "$TARGET_JSON" \
      --audit-check-to-actions-mapping "$AUDIT_CHECK_MAPPING" --output json 2>&1)
 || true

    if ! check_error "$MITIGATION_TASK_RESULT"; then
        echo "WARNING: Failed to start mitigation actions task, continuing..."
    else
        echo "Mitigation actions task started successfully"
        CREATED_RESOURCES+=("Mitigation Task: $MITIGATION_TASK_ID")
    fi
else
    echo ""
    echo "=================================================="
    echo "Step 5: Skipping mitigation action application (no findings)"
    echo "=================================================="
fi

# Step 6: Set up SNS notifications (optional)
echo ""
echo "=================================================="
echo "Step 6: Setting up SNS notifications"
echo "=================================================="

# Check if SNS topic already exists
SNS_TOPICS=$(aws sns list-topics --output json 2>&1) || true
TOPIC_ARN=""
if validate_json "$SNS_TOPICS"; then
    TOPIC_ARN=$(echo "$SNS_TOPICS" | jq -r '.Topics[] | select(.TopicArn |
 contains("IoTDDNotifications")) | .TopicArn' 2>/dev/null | head -1 || echo "")
fi

if [ -n "$TOPIC_ARN" ]; then
    echo "SNS topic 'IoTDDNotifications' already exists, using existing topic..."
    echo "Topic ARN: $TOPIC_ARN"
else
    echo "Creating SNS topic for notifications..."
    SNS_RESULT=$(aws sns create-topic --name "IoTDDNotifications" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-iot-device-defender-gs --
output json 2>&1) || true
```

```
    if ! check_error "$SNS_RESULT"; then
        echo "WARNING: Failed to create SNS topic, continuing..."
        SNS_RESULT=""
    else
        TOPIC_ARN=$(extract_json_value "$SNS_RESULT" "TopicArn")
        if [ -n "$TOPIC_ARN" ]; then
            echo "SNS topic created with ARN: $TOPIC_ARN"
            CREATED_RESOURCES+=("SNS Topic: IoTDDNotifications")
        fi
    fi
fi

if [ -n "$TOPIC_ARN" ]; then
    echo "Updating audit configuration to enable SNS notifications..."

    # Build SNS notification configuration JSON
    SNS_CONFIG=$(cat <<EOF
{
  "SNS": {
    "targetArn": "$TOPIC_ARN",
    "roleArn": "$AUDIT_ROLE_ARN",
    "enabled": true
  }
}
EOF
)

    if ! validate_json "$SNS_CONFIG"; then
        echo "ERROR: Invalid SNS configuration JSON"
        exit 1
    fi

    SNS_UPDATE_RESULT=$(aws iot update-account-audit-configuration \
      --audit-notification-target-configurations "$SNS_CONFIG" 2>&1) || true

    if ! check_error "$SNS_UPDATE_RESULT"; then
        echo "WARNING: Failed to update audit configuration for SNS
 notifications"
    else
        echo "SNS notifications enabled successfully"
    fi
else
    echo "Skipping SNS configuration due to topic creation failure"
```

```
fi

# Step 7: Enable AWS IoT logging
echo ""
echo "================================================="
echo "Step 7: Enabling AWS IoT logging"
echo "================================================="

echo "Setting up AWS IoT logging options..."

LOGGING_RESULT=$(aws iot set-v2-logging-options \
  --role-arn "$LOGGING_ROLE_ARN" \
  --default-log-level "ERROR" 2>&1) || true

if ! check_error "$LOGGING_RESULT"; then
    echo "V2 logging setup failed, trying v1 logging..."

    V1_LOGGING_CONFIG=$(cat <<EOF
{
  "roleArn": "$LOGGING_ROLE_ARN",
  "logLevel": "ERROR"
}
EOF
)

    if ! validate_json "$V1_LOGGING_CONFIG"; then
        echo "ERROR: Invalid v1 logging configuration JSON"
        exit 1
    fi

    LOGGING_RESULT_V1=$(aws iot set-logging-options \
      --logging-options-payload "$V1_LOGGING_CONFIG" 2>&1) || true

    if ! check_error "$LOGGING_RESULT_V1"; then
        echo "WARNING: Failed to set up AWS IoT logging with both v1 and v2
 methods, continuing..."
    else
        echo "AWS IoT v1 logging enabled successfully"
    fi
else
    echo "AWS IoT v2 logging enabled successfully"
fi

# Verify logging is enabled
```

```
echo "Verifying logging configuration..."
LOGGING_CONFIG=$(aws iot get-v2-logging-options --output json 2>&1) || true
if [ -n "$LOGGING_CONFIG" ] && ! check_error "$LOGGING_CONFIG" && validate_json
 "$LOGGING_CONFIG"; then
    echo "Logging configuration verified"
    echo "$LOGGING_CONFIG" | jq '.' 2>/dev/null || echo "Configuration retrieved
 but could not display details"
else
    echo "Could not verify logging configuration, but setup completed"
fi

# Script completed successfully
echo ""
echo "================================================="
echo "AWS IoT Device Defender setup completed successfully!"
echo "================================================="
echo "The following resources were created:"
for resource in "${CREATED_RESOURCES[@]+"${CREATED_RESOURCES[@]}"}"; do
    echo "- $resource"
done
echo ""

# Cleanup phase
echo "=========================================="
echo "CLEANUP"
echo "=========================================="
echo "Starting automatic cleanup of resources..."
echo "Waiting 10 seconds before cleanup to allow resource stabilization..."
sleep 10

# Disable AWS IoT logging
echo "Disabling AWS IoT logging..."

DISABLE_V2_RESULT=$(aws iot set-v2-logging-options \
  --default-log-level "DISABLED" 2>&1) || true

if check_error "$DISABLE_V2_RESULT"; then
    echo "V2 logging disabled successfully"
else
    echo "Attempting v1 logging disable..."

    V1_DISABLE_CONFIG=$(cat <<'EOF'
{
  "logLevel": "DISABLED"
```

```
}
EOF
)

    DISABLE_V1_RESULT=$(aws iot set-logging-options \
      --logging-options-payload "$V1_DISABLE_CONFIG" 2>&1) || true

    if check_error "$DISABLE_V1_RESULT"; then
        echo "V1 logging disabled successfully"
    else
        echo "WARNING: Could not disable logging"
    fi
fi

# Delete mitigation action
echo "Deleting mitigation action..."
aws iot delete-mitigation-action --action-name "EnableErrorLoggingAction" 2>&1 ||
 true

# Reset audit configuration
echo "Resetting IoT Device Defender audit configuration..."
RESET_AUDIT_CONFIG='{"LOGGING_DISABLED_CHECK":{"enabled":false}}'
aws iot update-account-audit-configuration \
  --audit-check-configurations "$RESET_AUDIT_CONFIG" 2>&1 || true

# Delete SNS topic
echo "Deleting SNS topic..."
if [ -n "${TOPIC_ARN:-}" ] && [ "$TOPIC_ARN" != "null" ]; then
    aws sns delete-topic --topic-arn "$TOPIC_ARN" 2>&1 || true
fi

# Clean up IAM roles with improved error handling
echo "Cleaning up IAM roles..."

cleanup_role() {
    local role_name=$1
    echo "Cleaning up role: $role_name"

    if aws iam get-role --role-name "$role_name" >/dev/null 2>&1; then
        ROLE_POLICIES=$(aws iam list-role-policies --role-name "$role_name" --
output json 2>&1 || echo '{"PolicyNames":[]}')
        if validate_json "$ROLE_POLICIES"; then
            while IFS= read -r policy_name; do
                if [ -n "$policy_name" ] && [ "$policy_name" != "null" ]; then
```

```
                    echo "  Deleting inline policy: $policy_name"
                    aws iam delete-role-policy \
                        --role-name "$role_name" \
                        --policy-name "$policy_name" 2>&1 || true
                fi
            done < <(echo "$ROLE_POLICIES" | jq -r '.PolicyNames[]' 2>/dev/null
 || echo "")
        fi

        ATTACHED_POLICIES=$(aws iam list-attached-role-policies --role-name
 "$role_name" --output json 2>&1 || echo '{"AttachedPolicies":[]}')
        if validate_json "$ATTACHED_POLICIES"; then
            while IFS= read -r policy_arn; do
                if [ -n "$policy_arn" ] && [ "$policy_arn" != "null" ]; then
                    echo "  Detaching managed policy: $policy_arn"
                    aws iam detach-role-policy \
                        --role-name "$role_name" \
                        --policy-arn "$policy_arn" 2>&1 || true
                fi
            done < <(echo "$ATTACHED_POLICIES" | jq -r
 '.AttachedPolicies[].PolicyArn' 2>/dev/null || echo "")
        fi

        echo "  Deleting role: $role_name"
        aws iam delete-role --role-name "$role_name" 2>&1 || true
    else
        echo "  Role $role_name does not exist or already deleted"
    fi
}

cleanup_role "AWSIoTDeviceDefenderAuditRole"
cleanup_role "AWSIoTLoggingRole"
cleanup_role "IoTMitigationActionErrorLoggingRole"

echo "Cleanup completed successfully"

echo ""
echo "Script execution completed at $(date)"
echo "Log file: $LOG_FILE"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy

- CreateMitigationAction
- CreateRole
- CreateTopic
- DeleteAccountAuditConfiguration
- DeleteMitigationAction
- DeleteRole
- DeleteRolePolicy
- DeleteTopic
- DescribeAccountAuditConfiguration
- DescribeAuditTask
- DetachRolePolicy
- GetLoggingOptions
- GetRole
- GetV2LoggingOptions
- ListAuditFindings
- ListAuditMitigationActionsTasks
- ListMitigationActions
- ListRolePolicies
- ListTopics
- PutRolePolicy
- SetLoggingOptions
- SetV2LoggingOptions
- StartAuditMitigationActionsTask
- StartOnDemandAuditTask
- UpdateAccountAuditConfiguration

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Commencer à utiliser les magasins de fonctionnalités d'apprentissage automatique

L'exemple de code suivant illustre comment :

- Définir des autorisations IAM
- Création d'un rôle SageMaker d'exécution
- Création de groupes de fonctionnalités
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Amazon SageMaker Feature Store Tutorial Script - Version 3
# This script demonstrates how to use Amazon SageMaker Feature Store with AWS CLI

# Setup logging
LOG_FILE="sagemaker-featurestore-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting SageMaker Feature Store tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"
echo ""

# Track created resources for cleanup
CREATED_RESOURCES=()

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
```

```
    exit 1
}

# Function to check command status
check_status() {
    if echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to wait for feature group to be created
wait_for_feature_group() {
    local feature_group_name=$1
    local status="Creating"

    echo "Waiting for feature group ${feature_group_name} to be created..."

    while [ "$status" = "Creating" ]; do
        sleep 5
        status=$(aws sagemaker describe-feature-group \
            --feature-group-name "${feature_group_name}" \
            --query 'FeatureGroupStatus' \
            --output text)
        echo "Current status: ${status}"

        if [ "$status" = "Failed" ]; then
            handle_error "Feature group ${feature_group_name} creation failed"
        fi
    done

    echo "Feature group ${feature_group_name} is now ${status}"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Clean up in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[$i]}"
        resource_type=$(echo "$resource" | cut -d: -f1)
        resource_name=$(echo "$resource" | cut -d: -f2)

        echo "Deleting $resource_type: $resource_name"
```

```
        case "$resource_type" in
            "FeatureGroup")
                aws sagemaker delete-feature-group --feature-group-name
 "$resource_name"
                ;;
            "S3Bucket")
                echo "Emptying S3 bucket: $resource_name"
                aws s3 rm "s3://$resource_name" --recursive 2>/dev/null
                echo "Deleting S3 bucket: $resource_name"
                aws s3api delete-bucket --bucket "$resource_name" 2>/dev/null
                ;;
            "IAMRole")
                echo "Detaching policies from role: $resource_name"
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>/dev/null
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>/dev/null
                echo "Deleting IAM role: $resource_name"
                aws iam delete-role --role-name "$resource_name" 2>/dev/null
                ;;
            *)
                echo "Unknown resource type: $resource_type"
                ;;
        esac
    done
}

# Function to create SageMaker execution role
create_sagemaker_role() {
    local role_name="SageMakerFeatureStoreRole-$(openssl rand -hex 4)"

    echo "Creating SageMaker execution role: $role_name" >&2

    # Create trust policy document
    local trust_policy='{
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "Service": "sagemaker.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
```

```
            }
        ]
    }'

    # Create the role
    local role_result=$(aws iam create-role \
        --role-name "$role_name" \
        --assume-role-policy-document "$trust_policy" \
        --description "SageMaker execution role for Feature Store tutorial" 2>&1)

    if echo "$role_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to create IAM role: $role_result"
    fi

    echo "Role created successfully" >&2
    CREATED_RESOURCES+=("IAMRole:$role_name")

    # Tag the role
    echo "Tagging IAM role..." >&2
    aws iam tag-role --role-name "$role_name" --tags Key=project,Value=doc-smith
Key=tutorial,Value=sagemaker-featurestore 2>&1

    # Attach necessary policies
    echo "Attaching policies to role..." >&2

    # SageMaker execution policy
    local policy1_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>&1)

    if echo "$policy1_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to attach SageMaker policy: $policy1_result"
    fi

    # S3 access policy
    local policy2_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>&1)

    if echo "$policy2_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to attach S3 policy: $policy2_result"
    fi

    # Get account ID for role ARN
```

```
    local account_id=$(aws sts get-caller-identity --query Account --output text)
    local role_arn="arn:aws:iam::${account_id}:role/${role_name}"

    echo "Role ARN: $role_arn" >&2
    echo "Waiting 10 seconds for role to propagate..." >&2
    sleep 10

    # Return only the role ARN to stdout
    echo "$role_arn"
}

# Handle SageMaker execution role
ROLE_ARN=""

if [ -z "$1" ]; then
    echo "Creating SageMaker execution role automatically..."
    ROLE_ARN=$(create_sagemaker_role)
    if [ -z "$ROLE_ARN" ]; then
        handle_error "Failed to create SageMaker execution role"
    fi
else
    ROLE_ARN="$1"

    # Validate the role ARN
    ROLE_NAME=$(echo "$ROLE_ARN" | sed 's/.*role\///')
    ROLE_CHECK=$(aws iam get-role --role-name "$ROLE_NAME" 2>&1)
    if echo "$ROLE_CHECK" | grep -i "error" > /dev/null; then
        echo "Creating a new role automatically..."
        ROLE_ARN=$(create_sagemaker_role)
        if [ -z "$ROLE_ARN" ]; then
            handle_error "Failed to create SageMaker execution role"
        fi
    fi
fi

# Handle cleanup option
AUTO_CLEANUP=""
if [ -n "$2" ]; then
    AUTO_CLEANUP="$2"
fi

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
echo "Using random identifier: $RANDOM_ID"
```

```
# Set variables
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
check_status "$ACCOUNT_ID"
echo "Account ID: $ACCOUNT_ID"

# Get current region
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No default region configured, using: $REGION"
else
    echo "Using region: $REGION"
fi
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="sagemaker-featurestore-${RANDOM_ID}-${ACCOUNT_ID}"
fi
PREFIX="featurestore-tutorial"
CURRENT_TIME=$(date +%s)

echo "Creating S3 bucket: $S3_BUCKET_NAME"
# Create bucket in current region (skip if using shared bucket)
if [ "$BUCKET_IS_SHARED" = "false" ]; then
if [ "$REGION" = "us-east-1" ]; then
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" 2>&1)
else
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" \
        --create-bucket-configuration LocationConstraint="$REGION" 2>&1)
fi

if echo "$BUCKET_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to create S3 bucket: $BUCKET_RESULT"
```

```
    exit 1
fi

echo "$BUCKET_RESULT"
CREATED_RESOURCES+=("S3Bucket:$S3_BUCKET_NAME")

# Tag the S3 bucket
echo "Tagging S3 bucket: $S3_BUCKET_NAME"
aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=sagemaker-
featurestore}]' 2>&1

# Block public access to the bucket
BLOCK_RESULT=$(aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
 2>&1)

if echo "$BLOCK_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to block public access to S3 bucket: $BLOCK_RESULT"
    cleanup_resources
    exit 1
fi
else
    echo "Using shared bucket (skipping creation)"
fi

# Create feature groups
echo "Creating feature groups..."

# Create customers feature group
CUSTOMERS_FEATURE_GROUP_NAME="customers-feature-group-${RANDOM_ID}"
echo "Creating customers feature group: $CUSTOMERS_FEATURE_GROUP_NAME"

CUSTOMERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "name", "FeatureType": "String"},
        {"FeatureName": "age", "FeatureType": "Integral"},
        {"FeatureName": "address", "FeatureType": "String"},
```

```
        {"FeatureName": "membership_type", "FeatureType": "String"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
        },
        "DisableGlueTableCreation": false
    }' \
    --role-arn "$ROLE_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$CUSTOMERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create customers feature group: $CUSTOMERS_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$CUSTOMERS_FEATURE_GROUP_NAME")

# Create orders feature group
ORDERS_FEATURE_GROUP_NAME="orders-feature-group-${RANDOM_ID}"
echo "Creating orders feature group: $ORDERS_FEATURE_GROUP_NAME"

ORDERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "order_id", "FeatureType": "String"},
        {"FeatureName": "order_date", "FeatureType": "String"},
        {"FeatureName": "product", "FeatureType": "String"},
        {"FeatureName": "quantity", "FeatureType": "Integral"},
        {"FeatureName": "amount", "FeatureType": "Fractional"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
```

```
        },
        "DisableGlueTableCreation": false
    }' \
    --role-arn "$ROLE_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$ORDERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create orders feature group: $ORDERS_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$ORDERS_FEATURE_GROUP_NAME")

# Wait for feature groups to be created
wait_for_feature_group "$CUSTOMERS_FEATURE_GROUP_NAME"
wait_for_feature_group "$ORDERS_FEATURE_GROUP_NAME"

# Ingest data into feature groups
echo "Ingesting data into feature groups..."

# Ingest customer data
echo "Ingesting customer data..."
CUSTOMER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "name", "ValueAsString": "John Doe"},
        {"FeatureName": "age", "ValueAsString": "35"},
        {"FeatureName": "address", "ValueAsString": "123 Main St"},
        {"FeatureName": "membership_type", "ValueAsString": "premium"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 1 data: $CUSTOMER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER1_RESPONSE"
```

```
CUSTOMER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "name", "ValueAsString": "Jane Smith"},
        {"FeatureName": "age", "ValueAsString": "28"},
        {"FeatureName": "address", "ValueAsString": "456 Oak Ave"},
        {"FeatureName": "membership_type", "ValueAsString": "standard"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 2 data: $CUSTOMER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER2_RESPONSE"

# Ingest order data
echo "Ingesting order data..."
ORDER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-001"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-15"},
        {"FeatureName": "product", "ValueAsString": "Laptop"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "1299.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 1 data: $ORDER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER1_RESPONSE"

ORDER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
```

```
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-002"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-20"},
        {"FeatureName": "product", "ValueAsString": "Smartphone"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "899.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 2 data: $ORDER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER2_RESPONSE"

# Retrieve records from feature groups
echo "Retrieving records from feature groups..."

# Get a single customer record
echo "Getting customer record with ID 573291:"
CUSTOMER_RECORD=$(aws sagemaker-featurestore-runtime get-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-value-as-string "573291" 2>&1)

if echo "$CUSTOMER_RECORD" | grep -i "error" > /dev/null; then
    echo "Failed to get customer record: $CUSTOMER_RECORD"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER_RECORD"

# Get multiple records using batch-get-record
echo "Getting multiple records using batch-get-record:"
BATCH_RECORDS=$(aws sagemaker-featurestore-runtime batch-get-record \
    --identifiers '[
        {
            "FeatureGroupName": "'${CUSTOMERS_FEATURE_GROUP_NAME}'",
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        },
        {
            "FeatureGroupName": "'${ORDERS_FEATURE_GROUP_NAME}'",
```

```
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        }
    ]' 2>&1)

if echo "$BATCH_RECORDS" | grep -i "error" > /dev/null && ! echo "$BATCH_RECORDS"
 | grep -i "Records" > /dev/null; then
    echo "Failed to get batch records: $BATCH_RECORDS"
    cleanup_resources
    exit 1
fi

echo "$BATCH_RECORDS"

# List feature groups
echo "Listing feature groups:"
FEATURE_GROUPS=$(aws sagemaker list-feature-groups 2>&1)

if echo "$FEATURE_GROUPS" | grep -i "error" > /dev/null; then
    echo "Failed to list feature groups: $FEATURE_GROUPS"
    cleanup_resources
    exit 1
fi

echo "$FEATURE_GROUPS"

# Display summary of created resources
echo ""
echo "============================================="
echo "TUTORIAL COMPLETED SUCCESSFULLY!"
echo "============================================="
echo "Resources created:"
echo "- S3 Bucket: $S3_BUCKET_NAME"
echo "- Customers Feature Group: $CUSTOMERS_FEATURE_GROUP_NAME"
echo "- Orders Feature Group: $ORDERS_FEATURE_GROUP_NAME"
if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
    echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o 'IAMRole:
[^[:space:]]*' | cut -d: -f2)"
fi
echo ""
echo "You can now:"
echo "1. View your feature groups in the SageMaker console"
echo "2. Query the offline store using Amazon Athena"
echo "3. Use the feature groups in your ML workflows"
echo "============================================="
```

```
echo ""

# Handle cleanup
if [ "$AUTO_CLEANUP" = "y" ]; then
    echo "Auto-cleanup enabled. Starting cleanup..."
    cleanup_resources
    echo "Cleanup completed."
elif [ "$AUTO_CLEANUP" = "n" ]; then
    echo "Auto-cleanup disabled. Resources will remain in your account."
    echo "To clean up later, run this script again with cleanup option 'y'"
else
    echo "==========================================="
    echo "CLEANUP CONFIRMATION"
    echo "==========================================="
    echo "Do you want to clean up all created resources? (y/n): "
    read -r CLEANUP_CHOICE

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        echo "Starting cleanup..."
        cleanup_resources
        echo "Cleanup completed."
    else
        echo "Skipping cleanup. Resources will remain in your account."
        echo "To clean up later, delete the following resources:"
        echo "- Feature Groups: $CUSTOMERS_FEATURE_GROUP_NAME,
 $ORDERS_FEATURE_GROUP_NAME"
        echo "- S3 Bucket: $S3_BUCKET_NAME"
        if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
            echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o
 'IAMRole:[^[:space:]]*' | cut -d: -f2)"
        fi
        echo ""
        echo "Estimated ongoing cost: ~$0.01 per month for online store"
    fi
fi

echo "Script completed at $(date)"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreateBucket

- CreateFeatureGroup
- CreateRole
- DeleteBucket
- DeleteFeatureGroup
- DeleteRole
- DescribeFeatureGroup
- DetachRolePolicy
- GetCallerIdentity
- GetRole
- ListFeatureGroups
- PutPublicAccessBlock
- Rm

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Commencer à utiliser les clusters Kubernetes gérés

L'exemple de code suivant illustre comment :

- Créez un VPC pour votre cluster EKS
- Créez des rôles IAM pour votre cluster EKS
- Créer votre cluster EKS
- Configurer kubectl pour communiquer avec votre cluster
- Créer un groupe de nœuds géré
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Amazon EKS Cluster Creation Script (v2)
# This script creates an Amazon EKS cluster with a managed node group using the
 AWS CLI

# Set up logging
LOG_FILE="eks-cluster-creation-v2.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon EKS cluster creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Error handling function
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check command success
check_command() {
    if [ $? -ne 0 ] || echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to check if kubectl is installed
check_kubectl() {
    if ! command -v kubectl &> /dev/null; then
        echo "WARNING: kubectl is not installed or not in your PATH."
```

```
        echo ""
        echo "To install kubectl, follow these instructions based on your
 operating system:"
        echo ""
        echo "For Linux:"
        echo "  1. Download the latest release:"
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl\""
        echo ""
        echo "  2. Make the kubectl binary executable:"
        echo "     chmod +x ./kubectl"
        echo ""
        echo "  3. Move the binary to your PATH:"
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For macOS:"
        echo "  1. Using Homebrew:"
        echo "     brew install kubectl"
        echo "     or"
        echo "  2. Using curl:"
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/darwin/amd64/kubectl\""
        echo "     chmod +x ./kubectl"
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For Windows:"
        echo "  1. Using curl:"
        echo "     curl -LO \"https://dl.k8s.io/release/v1.28.0/bin/windows/
amd64/kubectl.exe\""
        echo "     Add the binary to your PATH"
        echo "     or"
        echo "  2. Using Chocolatey:"
        echo "     choco install kubernetes-cli"
        echo ""
        echo "After installation, verify with: kubectl version --client"
        echo ""
        return 1
    fi
    return 0
}

# Generate a random identifier for resource names
RANDOM_ID=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 6 | head -n 1)
STACK_NAME="eks-vpc-stack-${RANDOM_ID}"
```

```
CLUSTER_NAME="eks-cluster-${RANDOM_ID}"
NODEGROUP_NAME="eks-nodegroup-${RANDOM_ID}"
CLUSTER_ROLE_NAME="EKSClusterRole-${RANDOM_ID}"
NODE_ROLE_NAME="EKSNodeRole-${RANDOM_ID}"

echo "Using the following resource names:"
echo "- VPC Stack: $STACK_NAME"
echo "- EKS Cluster: $CLUSTER_NAME"
echo "- Node Group: $NODEGROUP_NAME"
echo "- Cluster IAM Role: $CLUSTER_ROLE_NAME"
echo "- Node IAM Role: $NODE_ROLE_NAME"

# Array to track created resources for cleanup
declare -a CREATED_RESOURCES

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources in reverse order..."

    # Check if node group exists and delete it
    if aws eks list-nodegroups --cluster-name "$CLUSTER_NAME" --query
 "nodegroups[?contains(@,'$NODEGROUP_NAME')]" --output text 2>/dev/null | grep -q
 "$NODEGROUP_NAME"; then
        echo "Deleting node group: $NODEGROUP_NAME"
        aws eks delete-nodegroup --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
        echo "Waiting for node group deletion to complete..."
        aws eks wait nodegroup-deleted --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME"
        echo "Node group deleted successfully."
    fi

    # Check if cluster exists and delete it
    if aws eks describe-cluster --name "$CLUSTER_NAME" 2>/dev/null; then
        echo "Deleting cluster: $CLUSTER_NAME"
        aws eks delete-cluster --name "$CLUSTER_NAME"
        echo "Waiting for cluster deletion to complete (this may take several
 minutes)..."
        aws eks wait cluster-deleted --name "$CLUSTER_NAME"
        echo "Cluster deleted successfully."
    fi

    # Check if CloudFormation stack exists and delete it
```

```
    if aws cloudformation describe-stacks --stack-name "$STACK_NAME" 2>/dev/null;
 then
        echo "Deleting CloudFormation stack: $STACK_NAME"
        aws cloudformation delete-stack --stack-name "$STACK_NAME"
        echo "Waiting for CloudFormation stack deletion to complete..."
        aws cloudformation wait stack-delete-complete --stack-name "$STACK_NAME"
        echo "CloudFormation stack deleted successfully."
    fi

    # Clean up IAM roles
    if aws iam get-role --role-name "$NODE_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from node role: $NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSWorkerNodePolicy --role-name "$NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEC2ContainerRegistryReadOnly --role-name "$NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKS_CNI_Policy --role-name "$NODE_ROLE_NAME"
        echo "Deleting node role: $NODE_ROLE_NAME"
        aws iam delete-role --role-name "$NODE_ROLE_NAME"
        echo "Node role deleted successfully."
    fi

    if aws iam get-role --role-name "$CLUSTER_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from cluster role: $CLUSTER_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name "$CLUSTER_ROLE_NAME"
        echo "Deleting cluster role: $CLUSTER_ROLE_NAME"
        aws iam delete-role --role-name "$CLUSTER_ROLE_NAME"
        echo "Cluster role deleted successfully."
    fi

    echo "Cleanup complete."
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Cleaning up resources..."; cleanup_resources;
 exit 1' SIGINT SIGTERM

# Verify AWS CLI configuration
echo "Verifying AWS CLI configuration..."
AWS_ACCOUNT_INFO=$(aws sts get-caller-identity)
check_command "$AWS_ACCOUNT_INFO"
echo "AWS CLI is properly configured."
```

```
# Step 1: Create VPC using CloudFormation
echo "Step 1: Creating VPC with CloudFormation..."
echo "Creating CloudFormation stack: $STACK_NAME"

# Create the CloudFormation stack
CF_CREATE_OUTPUT=$(aws cloudformation create-stack \
  --stack-name "$STACK_NAME" \
  --template-url https://s3.us-west-2.amazonaws.com/amazon-eks/
cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs)
check_command "$CF_CREATE_OUTPUT"
CREATED_RESOURCES+=("CloudFormation Stack: $STACK_NAME")

echo "Waiting for CloudFormation stack to complete (this may take a few
 minutes)..."
aws cloudformation wait stack-create-complete --stack-name "$STACK_NAME"
if [ $? -ne 0 ]; then
    handle_error "CloudFormation stack creation failed"
fi
echo "CloudFormation stack created successfully."

# Step 2: Create IAM roles for EKS
echo "Step 2: Creating IAM roles for EKS..."

# Create cluster role trust policy
echo "Creating cluster role trust policy..."
cat > eks-cluster-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create cluster role
echo "Creating cluster IAM role: $CLUSTER_ROLE_NAME"
```

```
CLUSTER_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$CLUSTER_ROLE_NAME" \
  --assume-role-policy-document file://"eks-cluster-role-trust-policy.json")
check_command "$CLUSTER_ROLE_OUTPUT"
aws iam tag-role --role-name "$CLUSTER_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $CLUSTER_ROLE_NAME")

# Attach policy to cluster role
echo "Attaching EKS cluster policy to role..."
ATTACH_CLUSTER_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy \
  --role-name "$CLUSTER_ROLE_NAME")
check_command "$ATTACH_CLUSTER_POLICY_OUTPUT"

# Create node role trust policy
echo "Creating node role trust policy..."
cat > node-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create node role
echo "Creating node IAM role: $NODE_ROLE_NAME"
NODE_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$NODE_ROLE_NAME" \
  --assume-role-policy-document file://"node-role-trust-policy.json")
check_command "$NODE_ROLE_OUTPUT"
aws iam tag-role --role-name "$NODE_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $NODE_ROLE_NAME")

# Attach policies to node role
echo "Attaching EKS node policies to role..."
```

```
ATTACH_NODE_POLICY1_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY1_OUTPUT"

ATTACH_NODE_POLICY2_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY2_OUTPUT"

ATTACH_NODE_POLICY3_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY3_OUTPUT"

# Step 3: Get VPC and subnet information
echo "Step 3: Getting VPC and subnet information..."

VPC_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='VpcId'].OutputValue" \
  --output text)
if [ -z "$VPC_ID" ]; then
    handle_error "Failed to get VPC ID from CloudFormation stack"
fi
echo "VPC ID: $VPC_ID"

SUBNET_IDS=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SubnetIds'].OutputValue" \
  --output text)
if [ -z "$SUBNET_IDS" ]; then
    handle_error "Failed to get Subnet IDs from CloudFormation stack"
fi
echo "Subnet IDs: $SUBNET_IDS"

SECURITY_GROUP_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SecurityGroups'].OutputValue" \
  --output text)
if [ -z "$SECURITY_GROUP_ID" ]; then
    handle_error "Failed to get Security Group ID from CloudFormation stack"
fi
echo "Security Group ID: $SECURITY_GROUP_ID"
```

```
# Step 4: Create EKS cluster
echo "Step 4: Creating EKS cluster: $CLUSTER_NAME"

CLUSTER_ROLE_ARN=$(aws iam get-role --role-name "$CLUSTER_ROLE_NAME" --query
 "Role.Arn" --output text)
if [ -z "$CLUSTER_ROLE_ARN" ]; then
    handle_error "Failed to get Cluster Role ARN"
fi

echo "Creating EKS cluster (this will take 10-15 minutes)..."
CREATE_CLUSTER_OUTPUT=$(aws eks create-cluster \
  --name "$CLUSTER_NAME" \
  --role-arn "$CLUSTER_ROLE_ARN" \
  --resources-vpc-config
 subnetIds="$SUBNET_IDS",securityGroupIds="$SECURITY_GROUP_ID" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_CLUSTER_OUTPUT"
CREATED_RESOURCES+=("EKS Cluster: $CLUSTER_NAME")

echo "Waiting for EKS cluster to become active (this may take 10-15 minutes)..."
aws eks wait cluster-active --name "$CLUSTER_NAME"
if [ $? -ne 0 ]; then
    handle_error "Cluster creation failed or timed out"
fi
echo "EKS cluster is now active."

# Step 5: Configure kubectl
echo "Step 5: Configuring kubectl to communicate with the cluster..."

# Check if kubectl is installed
if ! check_kubectl; then
    echo "Will skip kubectl configuration steps but continue with the script."
    echo "You can manually configure kubectl later with: aws eks update-
kubeconfig --name \"$CLUSTER_NAME\""
else
    UPDATE_KUBECONFIG_OUTPUT=$(aws eks update-kubeconfig --name "$CLUSTER_NAME")
    check_command "$UPDATE_KUBECONFIG_OUTPUT"
    echo "kubectl configured successfully."

    # Test kubectl configuration
    echo "Testing kubectl configuration..."
    KUBECTL_TEST_OUTPUT=$(kubectl get svc 2>&1)
    if [ $? -ne 0 ]; then
```

```
        echo "Warning: kubectl configuration test failed. This might be due to
 permissions or network issues."
        echo "Error details: $KUBECTL_TEST_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$KUBECTL_TEST_OUTPUT"
        echo "kubectl configuration test successful."
    fi
fi

# Step 6: Create managed node group
echo "Step 6: Creating managed node group: $NODEGROUP_NAME"

NODE_ROLE_ARN=$(aws iam get-role --role-name "$NODE_ROLE_NAME" --query "Role.Arn"
 --output text)
if [ -z "$NODE_ROLE_ARN" ]; then
    handle_error "Failed to get Node Role ARN"
fi

# Convert comma-separated subnet IDs to space-separated for the create-nodegroup
 command
SUBNET_IDS_ARRAY=(${SUBNET_IDS//,/ })

echo "Creating managed node group (this will take 5-10 minutes)..."
CREATE_NODEGROUP_OUTPUT=$(aws eks create-nodegroup \
  --cluster-name "$CLUSTER_NAME" \
  --nodegroup-name "$NODEGROUP_NAME" \
  --node-role "$NODE_ROLE_ARN" \
  --subnets "${SUBNET_IDS_ARRAY[@]}" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_NODEGROUP_OUTPUT"
CREATED_RESOURCES+=("EKS Node Group: $NODEGROUP_NAME")

echo "Waiting for node group to become active (this may take 5-10 minutes)..."
aws eks wait nodegroup-active --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
if [ $? -ne 0 ]; then
    handle_error "Node group creation failed or timed out"
fi
echo "Node group is now active."

# Step 7: Verify nodes
echo "Step 7: Verifying nodes..."
```

```
echo "Waiting for nodes to register with the cluster (this may take a few
 minutes)..."
sleep 60  # Give nodes more time to register

# Check if kubectl is installed before attempting to use it
if ! check_kubectl; then
    echo "Cannot verify nodes without kubectl. Skipping this step."
    echo "You can manually verify nodes after installing kubectl with: kubectl
 get nodes"
else
    NODES_OUTPUT=$(kubectl get nodes 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: Unable to get nodes. This might be due to permissions or
 the nodes are still registering."
        echo "Error details: $NODES_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$NODES_OUTPUT"
        echo "Nodes verified successfully."
    fi
fi

# Step 8: View resources
echo "Step 8: Viewing cluster resources..."

echo "Cluster information:"
CLUSTER_INFO=$(aws eks describe-cluster --name "$CLUSTER_NAME")
echo "$CLUSTER_INFO"

echo "Node group information:"
NODEGROUP_INFO=$(aws eks describe-nodegroup --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME")
echo "$NODEGROUP_INFO"

echo "Kubernetes resources:"
if ! check_kubectl; then
    echo "Cannot list Kubernetes resources without kubectl. Skipping this step."
    echo "You can manually list resources after installing kubectl with: kubectl
 get all --all-namespaces"
else
    KUBE_RESOURCES=$(kubectl get all --all-namespaces 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: Unable to get Kubernetes resources. This might be due to
 permissions."
```

```
        echo "Error details: $KUBE_RESOURCES"
        echo "Continuing with script execution..."
    else
        echo "$KUBE_RESOURCES"
    fi
fi

# Display summary of created resources
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource"
done
echo "==========================================="

# Prompt for cleanup
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    cleanup_resources
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run the following commands:"
    echo "1. Delete node group: aws eks delete-nodegroup --cluster-name
 $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "2. Wait for node group deletion: aws eks wait nodegroup-deleted --
cluster-name $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "3. Delete cluster: aws eks delete-cluster --name $CLUSTER_NAME"
    echo "4. Wait for cluster deletion: aws eks wait cluster-deleted --name
 $CLUSTER_NAME"
    echo "5. Delete CloudFormation stack: aws cloudformation delete-stack --
stack-name $STACK_NAME"
    echo "6. Detach and delete IAM roles for the node group and cluster"
fi

echo "Script completed at $(date)"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreateCluster
  - CreateNodegroup
  - CreateRole
  - CreateStack
  - DeleteCluster
  - DeleteNodegroup
  - DeleteRole
  - DeleteStack
  - DescribeCluster
  - DescribeNodegroup
  - DescribeStacks
  - DetachRolePolicy
  - GetCallerIdentity
  - GetRole
  - ListNodegroups
  - UpdateKubeconfig
  - Attendre (CloudFormation)
  - Attendre (Amazon EKS)

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Débuter avec le streaming géré

L'exemple de code suivant illustre comment :

- Création d'un cluster MSK

- Création d'autorisations IAM pour l'accès MSK
- Création d'une machine cliente
- Trouvez des courtiers Bootstrap
- Configuration de la machine cliente
- Création d'un sujet et de produce/consume données
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Amazon MSK Getting Started Tutorial Script - Version 8
# This script automates the steps in the Amazon MSK Getting Started tutorial
# It creates an MSK cluster, sets up IAM permissions, creates a client machine,
# and configures the client to interact with the cluster

# Set up logging
LOG_FILE="msk_tutorial_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon MSK Getting Started Tutorial Script - Version 8"
echo "Logging to $LOG_FILE"
echo "================================================"

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created so far:"
    if [ -n "$CLUSTER_ARN" ]; then echo "- MSK Cluster: $CLUSTER_ARN"; fi
    if [ -n "$POLICY_ARN" ]; then echo "- IAM Policy: $POLICY_ARN"; fi
    if [ -n "$ROLE_NAME" ]; then echo "- IAM Role: $ROLE_NAME"; fi
```

```
    if [ -n "$INSTANCE_PROFILE_NAME" ]; then echo "- IAM Instance Profile:
 $INSTANCE_PROFILE_NAME"; fi
    if [ -n "$CLIENT_SG_ID" ]; then echo "- Client Security Group:
 $CLIENT_SG_ID"; fi
    if [ -n "$INSTANCE_ID" ]; then echo "- EC2 Instance: $INSTANCE_ID"; fi
    if [ -n "$KEY_NAME" ]; then echo "- Key Pair: $KEY_NAME"; fi

    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check if a resource exists
resource_exists() {
    local resource_type="$1"
    local resource_id="$2"

    case "$resource_type" in
        "cluster")
            aws kafka describe-cluster --cluster-arn "$resource_id" &>/dev/null
            ;;
        "policy")
            aws iam get-policy --policy-arn "$resource_id" &>/dev/null
            ;;
        "role")
            aws iam get-role --role-name "$resource_id" &>/dev/null
            ;;
        "instance-profile")
            aws iam get-instance-profile --instance-profile-name "$resource_id"
 &>/dev/null
            ;;
        "security-group")
            aws ec2 describe-security-groups --group-ids "$resource_id" &>/dev/
null
            ;;
        "instance")
            aws ec2 describe-instances --instance-ids "$resource_id" --query
 'Reservations[0].Instances[0].State.Name' --output text | grep -v "terminated"
 &>/dev/null
            ;;
        "key-pair")
            aws ec2 describe-key-pairs --key-names "$resource_id" &>/dev/null
            ;;
    esac
```

```
}

# Function to remove security group references
remove_security_group_references() {
    local sg_id="$1"

    if [ -z "$sg_id" ]; then
        echo "No security group ID provided for reference removal"
        return
    fi

    echo "Removing security group references for $sg_id"

    # Get all security groups in the VPC that might reference our client security
 group
    local vpc_security_groups=$(aws ec2 describe-security-groups \
        --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" \
        --query 'SecurityGroups[].GroupId' \
        --output text 2>/dev/null)

    if [ -n "$vpc_security_groups" ]; then
        for other_sg in $vpc_security_groups; do
            if [ "$other_sg" != "$sg_id" ]; then
                echo "Checking security group $other_sg for references to $sg_id"

                # Get the security group details in JSON format
                local sg_details=$(aws ec2 describe-security-groups \
                    --group-ids "$other_sg" \
                    --output json 2>/dev/null)

                if [ -n "$sg_details" ]; then
                    # Check if our security group is referenced in inbound rules
                    local has_inbound_ref=$(echo "$sg_details" | grep -o
 "\"GroupId\": \"$sg_id\"" | head -1)

                    if [ -n "$has_inbound_ref" ]; then
                        echo "Found inbound rules in $other_sg referencing
 $sg_id, removing them..."

                        # Try to remove common rule types
                        echo "Attempting to remove all-traffic rule"
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol all \
```

```
                            --source-group "$sg_id" 2>/dev/null || echo "No all-
traffic rule to remove"

                        # Try to remove TCP rules on common ports
                        for port in 22 80 443 9092 9094 9096; do
                            aws ec2 revoke-security-group-ingress \
                                --group-id "$other_sg" \
                                --protocol tcp \
                                --port "$port" \
                                --source-group "$sg_id" 2>/dev/null || true
                        done

                        # Try to remove UDP rules
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol udp \
                            --source-group "$sg_id" 2>/dev/null || true
                    fi

                    # Check for outbound rules (less common but possible)
                    local has_outbound_ref=$(echo "$sg_details" | grep -A 20
 "IpPermissionsEgress" | grep -o "\"GroupId\": \"$sg_id\"" | head -1)

                    if [ -n "$has_outbound_ref" ]; then
                        echo "Found outbound rules in $other_sg referencing
 $sg_id, removing them..."

                        aws ec2 revoke-security-group-egress \
                            --group-id "$other_sg" \
                            --protocol all \
                            --source-group "$sg_id" 2>/dev/null || echo "No
 outbound all-traffic rule to remove"
                    fi
                fi
            fi
        done
    fi

    echo "Completed security group reference removal for $sg_id"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."
```

```
   # Delete EC2 instance if it exists
   if [ -n "$INSTANCE_ID" ] && resource_exists "instance" "$INSTANCE_ID"; then
       echo "Terminating EC2 instance: $INSTANCE_ID"
       aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || echo "Failed
to terminate instance"
       echo "Waiting for instance to terminate..."
       aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID" || echo
"Failed to wait for instance termination"
   fi

   # Delete MSK cluster first (to remove dependencies on security group)
   if [ -n "$CLUSTER_ARN" ] && resource_exists "cluster" "$CLUSTER_ARN"; then
       echo "Deleting MSK cluster: $CLUSTER_ARN"
       aws kafka delete-cluster --cluster-arn "$CLUSTER_ARN" || echo "Failed to
delete cluster"

       # Wait a bit for the cluster deletion to start
       echo "Waiting 30 seconds for cluster deletion to begin..."
       sleep 30
   fi

   # Remove security group references before attempting deletion
   if [ -n "$CLIENT_SG_ID" ] && resource_exists "security-group"
"$CLIENT_SG_ID"; then
       remove_security_group_references "$CLIENT_SG_ID"

       echo "Deleting security group: $CLIENT_SG_ID"
       # Try multiple times with longer delays to ensure dependencies are
removed
       for i in {1..10}; do
           if aws ec2 delete-security-group --group-id "$CLIENT_SG_ID"; then
               echo "Security group deleted successfully"
               break
           fi
           echo "Failed to delete security group (attempt $i/10), retrying in 30
seconds..."
           sleep 30
       done
   fi

   # Delete key pair if it exists
   if [ -n "$KEY_NAME" ] && resource_exists "key-pair" "$KEY_NAME"; then
       echo "Deleting key pair: $KEY_NAME"
```

```
        aws ec2 delete-key-pair --key-name "$KEY_NAME" || echo "Failed to delete
key pair"
    fi

    # Remove role from instance profile
    if [ -n "$ROLE_NAME" ] && [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists
"instance-profile" "$INSTANCE_PROFILE_NAME"; then
        echo "Removing role from instance profile"
        aws iam remove-role-from-instance-profile \
            --instance-profile-name "$INSTANCE_PROFILE_NAME" \
            --role-name "$ROLE_NAME" || echo "Failed to remove role from instance
profile"
    fi

    # Delete instance profile
    if [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists "instance-profile"
"$INSTANCE_PROFILE_NAME"; then
        echo "Deleting instance profile: $INSTANCE_PROFILE_NAME"
        aws iam delete-instance-profile \
            --instance-profile-name "$INSTANCE_PROFILE_NAME" || echo "Failed to
delete instance profile"
    fi

    # Detach policy from role
    if [ -n "$ROLE_NAME" ] && [ -n "$POLICY_ARN" ] && resource_exists "role"
"$ROLE_NAME"; then
        echo "Detaching policy from role"
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "$POLICY_ARN" || echo "Failed to detach policy"
    fi

    # Delete role
    if [ -n "$ROLE_NAME" ] && resource_exists "role" "$ROLE_NAME"; then
        echo "Deleting role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" || echo "Failed to delete
role"
    fi

    # Delete policy
    if [ -n "$POLICY_ARN" ] && resource_exists "policy" "$POLICY_ARN"; then
        echo "Deleting policy: $POLICY_ARN"
        aws iam delete-policy --policy-arn "$POLICY_ARN" || echo "Failed to
delete policy"
```

```
    fi

    echo "Cleanup completed"
}

# Function to find a suitable subnet and instance type combination
find_suitable_subnet_and_instance_type() {
    local vpc_id="$1"
    local -a subnet_array=("${!2}")

    # List of instance types to try, in order of preference
    local instance_types=("t3.micro" "t2.micro" "t3.small" "t2.small")

    echo "Finding suitable subnet and instance type combination..."

    for instance_type in "${instance_types[@]}"; do
        echo "Trying instance type: $instance_type"

        for subnet_id in "${subnet_array[@]}"; do
            # Get the availability zone for this subnet
            local az=$(aws ec2 describe-subnets \
                --subnet-ids "$subnet_id" \
                --query 'Subnets[0].AvailabilityZone' \
                --output text)

            echo "  Checking subnet $subnet_id in AZ $az"

            # Check if this instance type is available in this AZ
            local available=$(aws ec2 describe-instance-type-offerings \
                --location-type availability-zone \
                --filters "Name=location,Values=$az" "Name=instance-type,Values=
$instance_type" \
                --query 'InstanceTypeOfferings[0].InstanceType' \
                --output text 2>/dev/null)

            if [ "$available" = "$instance_type" ]; then
                echo "  # Found suitable combination: $instance_type in $az
 (subnet: $subnet_id)"
                SELECTED_SUBNET_ID="$subnet_id"
                SELECTED_INSTANCE_TYPE="$instance_type"
                return 0
            else
                echo "  # $instance_type not available in $az"
            fi
```

```
        done
    done

    echo "ERROR: Could not find any suitable subnet and instance type
 combination"
    return 1
}

# Generate unique identifiers
RANDOM_SUFFIX=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 8 | head -n 1)
CLUSTER_NAME="MSKTutorialCluster-${RANDOM_SUFFIX}"
POLICY_NAME="msk-tutorial-policy-${RANDOM_SUFFIX}"
ROLE_NAME="msk-tutorial-role-${RANDOM_SUFFIX}"
INSTANCE_PROFILE_NAME="msk-tutorial-profile-${RANDOM_SUFFIX}"
SG_NAME="MSKClientSecurityGroup-${RANDOM_SUFFIX}"

echo "Using the following resource names:"
echo "- Cluster Name: $CLUSTER_NAME"
echo "- Policy Name: $POLICY_NAME"
echo "- Role Name: $ROLE_NAME"
echo "- Instance Profile Name: $INSTANCE_PROFILE_NAME"
echo "- Security Group Name: $SG_NAME"
echo "================================================"

# Step 1: Create an MSK Provisioned cluster
echo "Step 1: Creating MSK Provisioned cluster"

# Get the default VPC ID first
echo "Getting default VPC..."
DEFAULT_VPC_ID=$(aws ec2 describe-vpcs \
    --filters "Name=is-default,Values=true" \
    --query "Vpcs[0].VpcId" \
    --output text)

if [ -z "$DEFAULT_VPC_ID" ] || [ "$DEFAULT_VPC_ID" = "None" ]; then
    handle_error "Could not find default VPC. Please ensure you have a default
 VPC in your region."
fi

echo "Default VPC ID: $DEFAULT_VPC_ID"

# Get available subnets in the default VPC
echo "Getting available subnets in the default VPC..."
SUBNETS=$(aws ec2 describe-subnets \
```

```
    --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" "Name=default-for-
az,Values=true" \
    --query "Subnets[0:3].SubnetId" \
    --output text)

# Convert space-separated subnet IDs to an array
read -r -a SUBNET_ARRAY <<< "$SUBNETS"

if [ ${#SUBNET_ARRAY[@]} -lt 3 ]; then
    handle_error "Not enough subnets available in the default VPC. Need at least
 3 subnets, found ${#SUBNET_ARRAY[@]}."
fi

# Get default security group for the default VPC
echo "Getting default security group for the default VPC..."
DEFAULT_SG=$(aws ec2 describe-security-groups \
    --filters "Name=group-name,Values=default" "Name=vpc-id,Values=
$DEFAULT_VPC_ID" \
    --query "SecurityGroups[0].GroupId" \
    --output text)

if [ -z "$DEFAULT_SG" ] || [ "$DEFAULT_SG" = "None" ]; then
    handle_error "Could not find default security group for VPC $DEFAULT_VPC_ID"
fi

echo "Creating MSK cluster: $CLUSTER_NAME"
echo "Using VPC: $DEFAULT_VPC_ID"
echo "Using subnets: ${SUBNET_ARRAY[0]} ${SUBNET_ARRAY[1]} ${SUBNET_ARRAY[2]}"
echo "Using security group: $DEFAULT_SG"

# Create the MSK cluster with proper error handling
CLUSTER_RESPONSE=$(aws kafka create-cluster \
    --cluster-name "$CLUSTER_NAME" \
    --broker-node-group-info "{\"InstanceType\": \"kafka.t3.small\",
 \"ClientSubnets\": [\"${SUBNET_ARRAY[0]}\", \"${SUBNET_ARRAY[1]}\",
 \"${SUBNET_ARRAY[2]}\"], \"SecurityGroups\": [\"$DEFAULT_SG\"]}" \
    --kafka-version "3.6.0" \
    --number-of-broker-nodes 3 \
    --encryption-info "{\"EncryptionInTransit\": {\"InCluster\": true,
 \"ClientBroker\": \"TLS\"}}" \
    --tags project=doc-smith,tutorial=amazon-managed-streaming-for-apache-kafka-
gs 2>&1)

# Check if the command was successful
```

```
if [ $? -ne 0 ]; then
    handle_error "Failed to create MSK cluster: $CLUSTER_RESPONSE"
fi

# Extract the cluster ARN using grep
CLUSTER_ARN=$(echo "$CLUSTER_RESPONSE" | grep -o '"ClusterArn": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLUSTER_ARN" ]; then
    handle_error "Failed to extract cluster ARN from response: $CLUSTER_RESPONSE"
fi

echo "MSK cluster creation initiated. ARN: $CLUSTER_ARN"
echo "Waiting for cluster to become active (this may take 15-20 minutes)..."

# Wait for the cluster to become active
while true; do
    CLUSTER_STATUS=$(aws kafka describe-cluster --cluster-arn "$CLUSTER_ARN" --
query "ClusterInfo.State" --output text 2>/dev/null)

    if [ $? -ne 0 ]; then
        echo "Failed to get cluster status. Retrying in 30 seconds..."
        sleep 30
        continue
    fi

    echo "Current cluster status: $CLUSTER_STATUS"

    if [ "$CLUSTER_STATUS" = "ACTIVE" ]; then
        echo "Cluster is now active!"
        break
    elif [ "$CLUSTER_STATUS" = "FAILED" ]; then
        handle_error "Cluster creation failed"
    fi

    echo "Still waiting for cluster to become active... (checking again in 60
 seconds)"
    sleep 60
done

echo "============================================="
# Step 2: Create an IAM role granting access to create topics on the Amazon MSK
 cluster
echo "Step 2: Creating IAM policy and role"
```

```
# Get account ID and region
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION=$(aws ec2 describe-availability-zones --query
 'AvailabilityZones[0].RegionName' --output text)
fi

if [ -z "$ACCOUNT_ID" ] || [ -z "$REGION" ]; then
    handle_error "Could not determine AWS account ID or region"
fi

echo "Account ID: $ACCOUNT_ID"
echo "Region: $REGION"

# Create IAM policy
echo "Creating IAM policy: $POLICY_NAME"
POLICY_DOCUMENT="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:Connect\",
                \"kafka-cluster:AlterCluster\",
                \"kafka-cluster:DescribeCluster\"
            ],
            \"Resource\": [
                \"$CLUSTER_ARN\"
            ]
        },
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:*Topic*\",
                \"kafka-cluster:WriteData\",
                \"kafka-cluster:ReadData\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:topic/$CLUSTER_NAME/*\"
            ]
        },
        {
```

```
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:AlterGroup\",
                \"kafka-cluster:DescribeGroup\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:group/$CLUSTER_NAME/*\"
            ]
        }
    ]
}"

POLICY_RESPONSE=$(aws iam create-policy \
    --policy-name "$POLICY_NAME" \
    --policy-document "$POLICY_DOCUMENT" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create IAM policy: $POLICY_RESPONSE"
fi

# Extract the policy ARN using grep
POLICY_ARN=$(echo "$POLICY_RESPONSE" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)

if [ -z "$POLICY_ARN" ]; then
    handle_error "Failed to extract policy ARN from response: $POLICY_RESPONSE"
fi

aws iam tag-policy --policy-arn "$POLICY_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs

echo "IAM policy created. ARN: $POLICY_ARN"

# Create IAM role for EC2
echo "Creating IAM role: $ROLE_NAME"
TRUST_POLICY="{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",
\"Principal\":{\"Service\":\"ec2.amazonaws.com\"},\"Action\":\"sts:AssumeRole
\"}]}"

ROLE_RESPONSE=$(aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document "$TRUST_POLICY" 2>&1)
```

```
# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create IAM role: $ROLE_RESPONSE"
fi

echo "IAM role created: $ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs

# Attach policy to role
echo "Attaching policy to role"
ATTACH_RESPONSE=$(aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "$POLICY_ARN" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to attach policy to role: $ATTACH_RESPONSE"
fi

echo "Policy attached to role"

# Create instance profile and add role to it
echo "Creating instance profile: $INSTANCE_PROFILE_NAME"
PROFILE_RESPONSE=$(aws iam create-instance-profile \
    --instance-profile-name "$INSTANCE_PROFILE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create instance profile: $PROFILE_RESPONSE"
fi

echo "Instance profile created"

echo "Adding role to instance profile"
ADD_ROLE_RESPONSE=$(aws iam add-role-to-instance-profile \
    --instance-profile-name "$INSTANCE_PROFILE_NAME" \
    --role-name "$ROLE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to add role to instance profile: $ADD_ROLE_RESPONSE"
fi
```

```
echo "Role added to instance profile"

# Wait a moment for IAM propagation
echo "Waiting 10 seconds for IAM propagation..."
sleep 10

echo "============================================="

# Step 3: Create a client machine
echo "Step 3: Creating client machine"

# Find a suitable subnet and instance type combination
if ! find_suitable_subnet_and_instance_type "$DEFAULT_VPC_ID" SUBNET_ARRAY[@];
 then
    handle_error "Could not find a suitable subnet and instance type combination"
fi

echo "Selected subnet: $SELECTED_SUBNET_ID"
echo "Selected instance type: $SELECTED_INSTANCE_TYPE"

# Verify the subnet is in the same VPC we're using
SUBNET_VPC_ID=$(aws ec2 describe-subnets \
    --subnet-ids "$SELECTED_SUBNET_ID" \
    --query 'Subnets[0].VpcId' \
    --output text)

if [ "$SUBNET_VPC_ID" != "$DEFAULT_VPC_ID" ]; then
    handle_error "Subnet VPC ($SUBNET_VPC_ID) does not match default VPC
 ($DEFAULT_VPC_ID)"
fi

echo "VPC ID: $SUBNET_VPC_ID"

# Get security group ID from the MSK cluster
echo "Getting security group ID from the MSK cluster"
MSK_SG_ID=$(aws kafka describe-cluster \
    --cluster-arn "$CLUSTER_ARN" \
    --query 'ClusterInfo.BrokerNodeGroupInfo.SecurityGroups[0]' \
    --output text)

if [ -z "$MSK_SG_ID" ] || [ "$MSK_SG_ID" = "None" ]; then
    handle_error "Failed to get security group ID from cluster"
fi
```

```
echo "MSK security group ID: $MSK_SG_ID"

# Create security group for client
echo "Creating security group for client: $SG_NAME"
CLIENT_SG_RESPONSE=$(aws ec2 create-security-group \
    --group-name "$SG_NAME" \
    --description "Security group for MSK client" \
    --vpc-id "$DEFAULT_VPC_ID" \
    --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs}]' 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create security group: $CLIENT_SG_RESPONSE"
fi

# Extract the security group ID using grep
CLIENT_SG_ID=$(echo "$CLIENT_SG_RESPONSE" | grep -o '"GroupId": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLIENT_SG_ID" ]; then
    handle_error "Failed to extract security group ID from response:
 $CLIENT_SG_RESPONSE"
fi

echo "Client security group created. ID: $CLIENT_SG_ID"

# Allow SSH access to client from your IP only
echo "Getting your public IP address"
MY_IP=$(curl -s https://checkip.amazonaws.com 2>/dev/null)

if [ -z "$MY_IP" ]; then
    echo "Warning: Could not determine your IP address. Using 0.0.0.0/0 (not
 recommended for production)"
    MY_IP="0.0.0.0/0"
else
    MY_IP="$MY_IP/32"
    echo "Your IP address: $MY_IP"
fi

echo "Adding SSH ingress rule to client security group"
SSH_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
```

```
    --group-id "$CLIENT_SG_ID" \
    --protocol tcp \
    --port 22 \
    --cidr "$MY_IP" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add SSH ingress rule: $SSH_RULE_RESPONSE"
    echo "You may need to manually add SSH access to security group
 $CLIENT_SG_ID"
fi

echo "SSH ingress rule added"

# Update MSK security group to allow traffic from client security group
echo "Adding ingress rule to MSK security group to allow traffic from client"
MSK_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
    --group-id "$MSK_SG_ID" \
    --protocol all \
    --source-group "$CLIENT_SG_ID" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add ingress rule to MSK security group:
 $MSK_RULE_RESPONSE"
    echo "You may need to manually add ingress rule to security group $MSK_SG_ID"
fi

echo "Ingress rule added to MSK security group"

# Create key pair
KEY_NAME="MSKKeyPair-${RANDOM_SUFFIX}"
echo "Creating key pair: $KEY_NAME"
KEY_RESPONSE=$(aws ec2 create-key-pair --key-name "$KEY_NAME" --tag-
specifications 'ResourceType=key-pair,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-managed-streaming-for-apache-kafka-gs}]' --query
 'KeyMaterial' --output text 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create key pair: $KEY_RESPONSE"
fi

# Save the private key to a file
```

```
KEY_FILE="${KEY_NAME}.pem"
echo "$KEY_RESPONSE" > "$KEY_FILE"
chmod 400 "$KEY_FILE"

echo "Key pair created and saved to $KEY_FILE"

# Get the latest Amazon Linux 2 AMI
echo "Getting latest Amazon Linux 2 AMI ID"
AMI_ID=$(aws ec2 describe-images \
    --owners amazon \
    --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
    --query "sort_by(Images, &CreationDate)[-1].ImageId" \
    --output text 2>/dev/null)

if [ -z "$AMI_ID" ] || [ "$AMI_ID" = "None" ]; then
    handle_error "Failed to get Amazon Linux 2 AMI ID"
fi

echo "Using AMI ID: $AMI_ID"

# Launch EC2 instance with the selected subnet and instance type
echo "Launching EC2 instance"
echo "Instance type: $SELECTED_INSTANCE_TYPE"
echo "Subnet: $SELECTED_SUBNET_ID"

INSTANCE_RESPONSE=$(aws ec2 run-instances \
    --image-id "$AMI_ID" \
    --instance-type "$SELECTED_INSTANCE_TYPE" \
    --key-name "$KEY_NAME" \
    --security-group-ids "$CLIENT_SG_ID" \
    --subnet-id "$SELECTED_SUBNET_ID" \
    --iam-instance-profile "Name=$INSTANCE_PROFILE_NAME" \
    --tag-specifications
 "ResourceType=instance,Tags=[{Key=Name,Value=MSKTutorialClient-
${RANDOM_SUFFIX}},{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-
managed-streaming-for-apache-kafka-gs}]" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to launch EC2 instance: $INSTANCE_RESPONSE"
fi

# Extract the instance ID using grep
```

```
INSTANCE_ID=$(echo "$INSTANCE_RESPONSE" | grep -o '"InstanceId": "[^"]*' | head
 -1 | cut -d'"' -f4)

if [ -z "$INSTANCE_ID" ]; then
    handle_error "Failed to extract instance ID from response:
 $INSTANCE_RESPONSE"
fi

echo "EC2 instance launched successfully. ID: $INSTANCE_ID"
echo "Waiting for instance to be running..."

# Wait for the instance to be running
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

if [ $? -ne 0 ]; then
    handle_error "Instance failed to reach running state"
fi

# Wait a bit more for the instance to initialize
echo "Instance is running. Waiting 30 seconds for initialization..."
sleep 30

# Get public DNS name of instance
CLIENT_DNS=$(aws ec2 describe-instances \
    --instance-ids "$INSTANCE_ID" \
    --query 'Reservations[0].Instances[0].PublicDnsName' \
    --output text)

if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
    echo "Warning: Could not get public DNS name for instance. Trying public
 IP..."
    CLIENT_DNS=$(aws ec2 describe-instances \
        --instance-ids "$INSTANCE_ID" \
        --query 'Reservations[0].Instances[0].PublicIpAddress' \
        --output text)

    if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
        handle_error "Failed to get public DNS name or IP address for instance"
    fi
fi

echo "Client instance DNS/IP: $CLIENT_DNS"
echo "============================================="
# Get bootstrap brokers with improved logic
```

```
echo "Getting bootstrap brokers"
MAX_RETRIES=10
RETRY_COUNT=0
BOOTSTRAP_BROKERS=""
AUTH_METHOD=""

while [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; do
    # Get the full bootstrap brokers response
    BOOTSTRAP_RESPONSE=$(aws kafka get-bootstrap-brokers \
        --cluster-arn "$CLUSTER_ARN" 2>/dev/null)

    if [ $? -eq 0 ] && [ -n "$BOOTSTRAP_RESPONSE" ]; then
        # Try to get IAM authentication brokers first using grep
        BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringSaslIam": "[^"]*' | cut -d'"' -f4)
        if [ -n "$BOOTSTRAP_BROKERS" ]; then
            AUTH_METHOD="IAM"
        else
            # Fall back to TLS authentication
            BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringTls": "[^"]*' | cut -d'"' -f4)
            if [ -n "$BOOTSTRAP_BROKERS" ]; then
                AUTH_METHOD="TLS"
            fi
        fi
    fi

    RETRY_COUNT=$((RETRY_COUNT + 1))

    if [ "$RETRY_COUNT" -ge "$MAX_RETRIES" ]; then
        echo "Warning: Could not get bootstrap brokers after $MAX_RETRIES
 attempts."
        echo "You may need to manually retrieve them later using:"
        echo "aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
        BOOTSTRAP_BROKERS="BOOTSTRAP_BROKERS_NOT_AVAILABLE"
        AUTH_METHOD="UNKNOWN"
        break
    fi

    if [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; then
        echo "Bootstrap brokers not available yet. Retrying in 30 seconds...
 (Attempt $RETRY_COUNT/$MAX_RETRIES)"
        sleep 30
    fi
```

```
done

echo "Bootstrap brokers: $BOOTSTRAP_BROKERS"
echo "Authentication method: $AUTH_METHOD"
echo "============================================="

# Create setup script for the client machine
echo "Creating setup script for the client machine"
cat > setup_client.sh << 'EOF'
#!/bin/bash

# Set up logging
LOG_FILE="client_setup_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting client setup"
echo "============================================="

# Install Java
echo "Installing Java"
sudo yum -y install java-11

# Set environment variables
echo "Setting up environment variables"
export KAFKA_VERSION="3.6.0"
echo "KAFKA_VERSION=$KAFKA_VERSION"

# Download and extract Apache Kafka
echo "Downloading Apache Kafka"
wget https://archive.apache.org/dist/kafka/$KAFKA_VERSION/kafka_2.13-
$KAFKA_VERSION.tgz
if [ $? -ne 0 ]; then
    echo "Failed to download Kafka. Trying alternative mirror..."
    wget https://www.apache.org/dyn/closer.cgi?path=/kafka/$KAFKA_VERSION/
kafka_2.13-$KAFKA_VERSION.tgz
fi

echo "Extracting Kafka"
tar -xzf kafka_2.13-$KAFKA_VERSION.tgz
export KAFKA_ROOT=$(pwd)/kafka_2.13-$KAFKA_VERSION
echo "KAFKA_ROOT=$KAFKA_ROOT"

# Download the MSK IAM authentication package (needed for both IAM and TLS)
echo "Downloading MSK IAM authentication package"
```

```
cd $KAFKA_ROOT/libs
wget https://github.com/aws/aws-msk-iam-auth/releases/latest/download/aws-msk-
iam-auth-1.1.6-all.jar
if [ $? -ne 0 ]; then
    echo "Failed to download specific version. Trying to get latest version..."
    LATEST_VERSION=$(curl -s https://api.github.com/repos/aws/aws-msk-iam-auth/
releases/latest | grep -o '"tag_name": "[^"]*' | cut -d'"' -f4)
    wget https://github.com/aws/aws-msk-iam-auth/releases/download/
$LATEST_VERSION/aws-msk-iam-auth-$LATEST_VERSION-all.jar
    if [ $? -ne 0 ]; then
        echo "Failed to download IAM auth package. Please check the URL and try
 again."
        exit 1
    fi
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-$LATEST_VERSION-all.jar
else
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
fi
echo "CLASSPATH=$CLASSPATH"

# Create client properties file based on authentication method
echo "Creating client properties file"
cd $KAFKA_ROOT/config

# The AUTH_METHOD_PLACEHOLDER will be replaced by the script
AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"

if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "Configuring for IAM authentication"
    cat > client.properties << 'EOT'
security.protocol=SASL_SSL
sasl.mechanism=AWS_MSK_IAM
sasl.jaas.config=software.amazon.msk.auth.iam.IAMLoginModule required;
sasl.client.callback.handler.class=software.amazon.msk.auth.iam.IAMClientCallbackHandler
EOT
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "Configuring for TLS authentication"
    cat > client.properties << 'EOT'
security.protocol=SSL
EOT
else
    echo "Unknown authentication method. Creating basic TLS configuration."
    cat > client.properties << 'EOT'
security.protocol=SSL
```

```
EOT
fi

echo "Client setup completed"
echo "============================================="

# Create a script to set up environment variables
cat > ~/setup_env.sh << 'EOT'
#!/bin/bash
export KAFKA_VERSION="3.6.0"
export KAFKA_ROOT=~/kafka_2.13-$KAFKA_VERSION
export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
export BOOTSTRAP_SERVER="BOOTSTRAP_SERVER_PLACEHOLDER"
export AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"

echo "Environment variables set:"
echo "KAFKA_VERSION=$KAFKA_VERSION"
echo "KAFKA_ROOT=$KAFKA_ROOT"
echo "CLASSPATH=$CLASSPATH"
echo "BOOTSTRAP_SERVER=$BOOTSTRAP_SERVER"
echo "AUTH_METHOD=$AUTH_METHOD"
EOT

chmod +x ~/setup_env.sh

echo "Created environment setup script: ~/setup_env.sh"
echo "Run 'source ~/setup_env.sh' to set up your environment"
EOF

# Replace placeholders in the setup script
if [ -n "$BOOTSTRAP_BROKERS" ] && [ "$BOOTSTRAP_BROKERS" != "None" ] &&
 [ "$BOOTSTRAP_BROKERS" != "BOOTSTRAP_BROKERS_NOT_AVAILABLE" ]; then
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|$BOOTSTRAP_BROKERS|g" setup_client.sh
else
    # If bootstrap brokers are not available, provide instructions to get them
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|\$(aws kafka get-bootstrap-brokers
 --cluster-arn $CLUSTER_ARN --query 'BootstrapBrokerStringTls' --output text)|g"
 setup_client.sh
fi

# Replace auth method placeholder
sed -i "s|AUTH_METHOD_PLACEHOLDER|$AUTH_METHOD|g" setup_client.sh

echo "Setup script created"
```

```
echo "============================================="

# Display summary of created resources
echo ""
echo "============================================="
echo "RESOURCE SUMMARY"
echo "============================================="
echo "MSK Cluster ARN: $CLUSTER_ARN"
echo "MSK Cluster Name: $CLUSTER_NAME"
echo "Authentication Method: $AUTH_METHOD"
echo "IAM Policy ARN: $POLICY_ARN"
echo "IAM Role Name: $ROLE_NAME"
echo "IAM Instance Profile: $INSTANCE_PROFILE_NAME"
echo "Client Security Group: $CLIENT_SG_ID"
echo "EC2 Instance ID: $INSTANCE_ID"
echo "EC2 Instance Type: $SELECTED_INSTANCE_TYPE"
echo "EC2 Instance DNS: $CLIENT_DNS"
echo "Key Pair: $KEY_NAME (saved to $KEY_FILE)"
echo "Bootstrap Brokers: $BOOTSTRAP_BROKERS"
echo "============================================="
echo ""

# Instructions for connecting to the instance and setting up the client
echo "NEXT STEPS:"
echo "1. Connect to your EC2 instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS"
echo ""
echo "2. Upload the setup script to your instance:"
echo "   scp -i $KEY_FILE setup_client.sh ec2-user@$CLIENT_DNS:~/"
echo ""
echo "3. Run the setup script on your instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS 'chmod +x ~/setup_client.sh && ~/
setup_client.sh'"
echo ""
echo "4. Source the environment setup script:"
echo "   source ~/setup_env.sh"
echo ""

# Provide different instructions based on authentication method
if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "5. Create a Kafka topic (using IAM authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
```

```
    echo "     --replication-factor 3 \\"
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "5. Create a Kafka topic (using TLS authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --replication-factor 3 \\"
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
else
    echo "5. Manually retrieve bootstrap brokers and configure authentication:"
    echo "   aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
fi

echo ""
echo "8. Verify the topic was created:"
```

```
echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --list \\"
echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
echo "     --command-config \$KAFKA_ROOT/config/client.properties"
echo "============================================="
echo ""

# Ask if user wants to clean up resources
echo ""
echo "============================================="
echo "CLEANUP CONFIRMATION"
echo "============================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ $CLEANUP_CHOICE =~ ^[Yy]$ ]]; then
    cleanup_resources
    echo "All resources have been cleaned up."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run this script again and choose 'y' when
 prompted."
fi

echo "Script completed successfully!"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateInstanceProfile
  - CreateKeyPair
  - CreatePolicy
  - CreateRole
  - CreateSecurityGroup
  - DeleteCluster

- DeleteInstanceProfile
- DeleteKeyPair
- DeletePolicy
- DeleteRole
- DeleteSecurityGroup
- DescribeAvailabilityZones
- DescribeCluster
- DescribeImages
- DescribeInstanceTypeOfferings
- DescribeInstances
- DescribeKeyPairs
- DescribeSecurityGroups
- DescribeSubnets
- DescribeVpcs
- DetachRolePolicy
- GetBootstrapBrokers
- GetCallerIdentity
- GetInstanceProfile
- GetPolicy
- GetRole
- RemoveRoleFromInstanceProfile
- RevokeSecurityGroupEgress
- RevokeSecurityGroupIngress
- RunInstances
- TerminateInstances
- Attente

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Commencer à utiliser des clusters d'entrepôts de données provisionnés

L'exemple de code suivant illustre comment :

- Création d'un cluster Redshift
- Création d'un rôle IAM pour l'accès à S3
- Création de tables et chargement de données
- Exécuter des exemples de requêtes
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Amazon Redshift Provisioned Cluster Tutorial Script
# This script creates a Redshift cluster, loads sample data, runs queries, and
 cleans up resources
# Version 4: Security improvements and best practices

set -euo pipefail

# Set up logging
LOG_FILE="redshift_tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Redshift tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to handle errors
handle_error() {
    echo "ERROR: $1" >&2
    echo "Resources created so far:"
```

```
    if [ -n "${CLUSTER_ID:-}" ]; then echo "- Redshift Cluster: $CLUSTER_ID"; fi
    if [ -n "${ROLE_NAME:-}" ]; then echo "- IAM Role: $ROLE_NAME"; fi

    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Delete the cluster if it exists
    if [ -n "${CLUSTER_ID:-}" ]; then
        echo "Deleting Redshift cluster: $CLUSTER_ID"
        aws redshift delete-cluster --cluster-identifier "$CLUSTER_ID" --skip-
final-cluster-snapshot 2>/dev/null || true
        echo "Waiting for cluster deletion to complete..."
        aws redshift wait cluster-deleted --cluster-identifier "$CLUSTER_ID" 2>/
dev/null || true
        echo "Cluster deleted successfully."
    fi

    # Delete the IAM role if it exists
    if [ -n "${ROLE_NAME:-}" ]; then
        echo "Removing IAM role policy..."
        aws iam delete-role-policy --role-name "$ROLE_NAME" --policy-name
 RedshiftS3Access 2>/dev/null || true

        echo "Deleting IAM role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi

    # Clean up temporary files
    rm -f redshift-trust-policy.json redshift-s3-policy.json

    echo "Cleanup completed."
}

# Trap errors and cleanup
trap 'handle_error "Script interrupted"' INT TERM

# Function to wait for SQL statement to complete
wait_for_statement() {
```

```
    local statement_id=$1
    local max_attempts=30
    local attempt=1
    local status=""

    echo "Waiting for statement $statement_id to complete..."

    while [ $attempt -le $max_attempts ]; do
        status=$(aws redshift-data describe-statement --id "$statement_id" --
query 'Status' --output text 2>/dev/null || echo "")

        if [ "$status" == "FINISHED" ]; then
            echo "Statement completed successfully."
            return 0
        elif [ "$status" == "FAILED" ]; then
            local error=$(aws redshift-data describe-statement --id
 "$statement_id" --query 'Error' --output text 2>/dev/null || echo "Unknown
 error")
            echo "Statement failed with error: $error" >&2
            return 1
        elif [ "$status" == "ABORTED" ]; then
            echo "Statement was aborted." >&2
            return 1
        fi

        echo "Statement status: $status. Waiting... (Attempt $attempt/
$max_attempts)"
        sleep 10
        ((attempt++))
    done

    echo "Timed out waiting for statement to complete." >&2
    return 1
}

# Function to check if IAM role is attached to cluster
check_role_attached() {
    local role_arn=$1
    local max_attempts=10
    local attempt=1

    echo "Checking if IAM role is attached to the cluster..."

    while [ $attempt -le $max_attempts ]; do
```

```
        local status=$(aws redshift describe-clusters \
            --cluster-identifier "$CLUSTER_ID" \
            --query "Clusters[0].IamRoles[?IamRoleArn=='$role_arn'].ApplyStatus"
 \
            --output text 2>/dev/null || echo "")

        if [ "$status" == "in-sync" ]; then
            echo "IAM role is successfully attached to the cluster."
            return 0
        fi

        echo "IAM role status: $status. Waiting... (Attempt $attempt/
$max_attempts)"
        sleep 30
        ((attempt++))
    done

    echo "Timed out waiting for IAM role to be attached." >&2
    return 1
}

# Validate required commands
for cmd in aws jq; do
    if ! command -v "$cmd" &> /dev/null; then
        handle_error "Required command '$cmd' not found. Please install it and
 try again."
    fi
done

# Validate AWS credentials
if ! aws sts get-caller-identity &>/dev/null; then
    handle_error "AWS credentials not configured or invalid"
fi

# Variables to track created resources
CLUSTER_ID="examplecluster"
ROLE_NAME="RedshiftS3Role-$(date +%s)"
DB_NAME="dev"
DB_USER="awsuser"

# Generate secure password using AWS Secrets Manager or random string
if command -v openssl &> /dev/null; then
    DB_PASSWORD=$(openssl rand -base64 32 | tr -d "=+/" | cut -c1-20)
else
```

```
    DB_PASSWORD="TempPass$(date +%s | md5sum | cut -c1-20)"
fi

# Validate password meets requirements
if [ ${#DB_PASSWORD} -lt 8 ]; then
    handle_error "Generated password does not meet minimum length requirement"
fi

# Store password securely (optional: use AWS Secrets Manager in production)
echo "Generated database password (store securely): $DB_PASSWORD"

echo "=== Step 1: Creating Amazon Redshift Cluster ==="

# Create the Redshift cluster with encryption and audit logging enabled
echo "Creating Redshift cluster: $CLUSTER_ID"
CLUSTER_RESULT=$(aws redshift create-cluster \
  --cluster-identifier "$CLUSTER_ID" \
  --node-type ra3.xlplus \
  --number-of-nodes 2 \
  --master-username "$DB_USER" \
  --master-user-password "$DB_PASSWORD" \
  --db-name "$DB_NAME" \
  --port 5439 \
  --encrypted \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-provisioned \
  2>&1) || handle_error "Failed to create Redshift cluster"

echo "$CLUSTER_RESULT"
echo "Waiting for cluster to become available..."

# Wait for the cluster to be available
aws redshift wait cluster-available --cluster-identifier "$CLUSTER_ID" ||
 handle_error "Timeout waiting for cluster to become available"

# Get cluster status to confirm
CLUSTER_STATUS=$(aws redshift describe-clusters \
  --cluster-identifier "$CLUSTER_ID" \
  --query 'Clusters[0].ClusterStatus' \
  --output text)

echo "Cluster status: $CLUSTER_STATUS"

echo "=== Step 2: Creating IAM Role for S3 Access ==="
```

```
# Create trust policy file with restricted permissions
echo "Creating trust policy for Redshift"
cat > redshift-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

chmod 600 redshift-trust-policy.json

# Create IAM role
echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
  --assume-role-policy-document file://redshift-trust-policy.json 2>&1) ||
 handle_error "Failed to create IAM role"

echo "$ROLE_RESULT"

# Get the role ARN
ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn' --output
 text)
echo "Role ARN: $ROLE_ARN"

# Tag the IAM role
echo "Tagging IAM role: $ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=redshift-provisioned

# Create policy document for S3 access with principle of least privilege
echo "Creating S3 access policy"
cat > redshift-s3-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::redshift-downloads",
        "arn:aws:s3:::redshift-downloads/*"
      ]
    }
  ]
}
EOF

chmod 600 redshift-s3-policy.json

# Attach policy to role
echo "Attaching S3 access policy to role"
POLICY_RESULT=$(aws iam put-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-name RedshiftS3Access \
  --policy-document file://redshift-s3-policy.json 2>&1) || handle_error "Failed
 to attach policy to role"

echo "$POLICY_RESULT"

# Attach role to cluster
echo "Attaching IAM role to Redshift cluster"
ATTACH_ROLE_RESULT=$(aws redshift modify-cluster-iam-roles \
  --cluster-identifier "$CLUSTER_ID" \
  --add-iam-roles "$ROLE_ARN" 2>&1) || handle_error "Failed to attach role to
 cluster"

echo "$ATTACH_ROLE_RESULT"

# Wait for the role to be attached
echo "Waiting for IAM role to be attached to the cluster..."
if ! check_role_attached "$ROLE_ARN"; then
    handle_error "Failed to attach IAM role to cluster"
fi

echo "=== Step 3: Getting Cluster Connection Information ==="
```

```
# Get cluster endpoint
CLUSTER_INFO=$(aws redshift describe-clusters \
  --cluster-identifier "$CLUSTER_ID" \
  --query 'Clusters[0].Endpoint.{Address:Address,Port:Port}' \
  --output json)

echo "Cluster endpoint information:"
echo "$CLUSTER_INFO"

echo "=== Step 4: Creating Tables and Loading Data ==="

echo "Creating sales table"
SALES_TABLE_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "DROP TABLE IF EXISTS sales; CREATE TABLE sales(salesid integer not null,
 listid integer not null distkey, sellerid integer not null, buyerid integer
 not null, eventid integer not null, dateid smallint not null sortkey, qtysold
 smallint not null, pricepaid decimal(8,2), commission decimal(8,2), saletime
 timestamp);" \
  --query 'Id' --output text)

echo "Sales table creation statement ID: $SALES_TABLE_ID"

# Wait for statement to complete
if ! wait_for_statement "$SALES_TABLE_ID"; then
    handle_error "Failed to create sales table"
fi

echo "Creating date table"
DATE_TABLE_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "DROP TABLE IF EXISTS date; CREATE TABLE date(dateid smallint not null
 distkey sortkey, caldate date not null, day character(3) not null, week smallint
 not null, month character(5) not null, qtr character(5) not null, year smallint
 not null, holiday boolean default('N'));" \
  --query 'Id' --output text)

echo "Date table creation statement ID: $DATE_TABLE_ID"

# Wait for statement to complete
```

```
if ! wait_for_statement "$DATE_TABLE_ID"; then
    handle_error "Failed to create date table"
fi

echo "Loading data into sales table"
SALES_LOAD_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "COPY sales FROM 's3://redshift-downloads/tickit/sales_tab.txt' DELIMITER
 '\t' TIMEFORMAT 'MM/DD/YYYY HH:MI:SS' REGION 'us-east-1' IAM_ROLE '$ROLE_ARN';"
 \
  --query 'Id' --output text)

echo "Sales data load statement ID: $SALES_LOAD_ID"

# Wait for statement to complete
if ! wait_for_statement "$SALES_LOAD_ID"; then
    handle_error "Failed to load data into sales table"
fi

echo "Loading data into date table"
DATE_LOAD_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "COPY date FROM 's3://redshift-downloads/tickit/date2008_pipe.txt'
 DELIMITER '|' REGION 'us-east-1' IAM_ROLE '$ROLE_ARN';" \
  --query 'Id' --output text)

echo "Date data load statement ID: $DATE_LOAD_ID"

# Wait for statement to complete
if ! wait_for_statement "$DATE_LOAD_ID"; then
    handle_error "Failed to load data into date table"
fi

echo "=== Step 5: Running Example Queries ==="

echo "Running query: Get definition for the sales table"
QUERY1_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
```

```
  --sql "SELECT * FROM pg_table_def WHERE tablename = 'sales';" \
  --query 'Id' --output text)

echo "Query 1 statement ID: $QUERY1_ID"

# Wait for statement to complete
if ! wait_for_statement "$QUERY1_ID"; then
    handle_error "Query 1 failed"
fi

# Get and display results
echo "Query 1 results (first 10 rows):"
aws redshift-data get-statement-result --id "$QUERY1_ID" --max-items 10

echo "Running query: Find total sales on a given calendar date"
QUERY2_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "SELECT sum(qtysold) FROM sales, date WHERE sales.dateid = date.dateid
 AND caldate = '2008-01-05';" \
  --query 'Id' --output text)

echo "Query 2 statement ID: $QUERY2_ID"

# Wait for statement to complete
if ! wait_for_statement "$QUERY2_ID"; then
    handle_error "Query 2 failed"
fi

# Get and display results
echo "Query 2 results:"
aws redshift-data get-statement-result --id "$QUERY2_ID"

echo "=== Tutorial Complete ==="
echo "The following resources were created:"
echo "- Redshift Cluster: $CLUSTER_ID"
echo "- IAM Role: $ROLE_NAME"

echo ""
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Cleaning up all created resources..."
```

```
cleanup_resources
echo "All resources have been cleaned up."

# Securely clear password from memory
DB_PASSWORD=""

echo "Script completed at $(date)"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - CreateCluster
  - CreateRole
  - DeleteCluster
  - DeleteRole
  - DeleteRolePolicy
  - DescribeClusters
  - GetRole
  - ModifyClusterIamRoles
  - PutRolePolicy
  - Attente

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Commencer à orchestrer les flux de travail

L'exemple de code suivant illustre comment :

- Création d'un rôle IAM pour Step Functions
- Créez votre première machine à états
- Démarrez l'exécution de votre machine à états
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# AWS Step Functions Getting Started Tutorial Script
# This script creates and runs a Step Functions state machine based on the AWS
 Step Functions Getting Started tutorial

set -euo pipefail

# Security: Restrict umask to prevent unintended file permissions
umask 077

# Parse command line arguments
AUTO_CLEANUP=true
while [[ $# -gt 0 ]]; do
    case $1 in
        --auto-cleanup)
            AUTO_CLEANUP=true
            shift
            ;;
        -h|--help)
            echo "Usage: $0 [--auto-cleanup] [--help]"
            echo "  --auto-cleanup: Automatically clean up resources without
 prompting"
            echo "  --help: Show this help message"
            exit 0
            ;;
        *)
            echo "Unknown option: $1"
            echo "Use --help for usage information"
            exit 1
            ;;
    esac
```

```
done

# Set up logging with secure permissions
LOG_FILE="step-functions-tutorial.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"

# Security: Use process substitution with explicit FD cleanup
exec 3>&1 4>&2
exec > >(tee -a "$LOG_FILE") 2>&1
trap 'exec 1>&3 2>&4 3>&- 4>&-' EXIT

echo "Starting AWS Step Functions Getting Started Tutorial..."
echo "Logging to $LOG_FILE"

# Verify AWS CLI is installed and configured
if ! command -v aws &> /dev/null; then
    echo "ERROR: AWS CLI is not installed"
    exit 1
fi

# Verify AWS credentials are configured
if ! aws sts get-caller-identity &> /dev/null; then
    echo "ERROR: AWS credentials are not configured or invalid"
    exit 1
fi

# Check if jq is available for better JSON parsing
if ! command -v jq &> /dev/null; then
    echo "WARNING: jq is not installed. Using basic JSON parsing which may be
 less reliable."
    echo "Consider installing jq for better error handling: brew install jq
 (macOS) or apt-get install jq (Ubuntu)"
    USE_JQ=false
else
    USE_JQ=true
fi

# Use fixed region that supports Amazon Comprehend
CURRENT_REGION="us-west-2"
echo "Using fixed AWS region: $CURRENT_REGION (supports Amazon Comprehend)"

# Set AWS CLI to use the fixed region for all commands
export AWS_DEFAULT_REGION="$CURRENT_REGION"
```

```
export AWS_REGION="$CURRENT_REGION"

# Amazon Comprehend is available in us-west-2, so we can always enable it
echo "Amazon Comprehend is available in region $CURRENT_REGION"
SKIP_COMPREHEND=false

# Security: Initialize all resource variables
STATE_MACHINE_ARN=""
ROLE_NAME=""
ROLE_ARN=""
POLICY_ARN=""
STEPFUNCTIONS_POLICY_ARN=""
EXECUTION_ARN=""
EXECUTION2_ARN=""
EXECUTION3_ARN=""

# Performance: Cache for AWS API calls to reduce redundant requests
declare -A API_CACHE

# Function to make cached AWS CLI calls
aws_call_cached() {
    local cache_key="$1"
    shift

    if [[ -v API_CACHE["$cache_key"] ]]; then
        echo "${API_CACHE[$cache_key]}"
        return 0
    fi

    local result
    result=$(aws "$@" 2>&1) || return $?
    API_CACHE["$cache_key"]="$result"
    echo "$result"
}

# Function to check for API errors in JSON response with optimized jq usage
check_api_error() {
    local response="$1"
    local operation="$2"

    if [[ "$USE_JQ" == "true" ]]; then
        # Use jq for more reliable JSON parsing with efficient error detection
        if echo "$response" | jq -e '.Error // .error // empty' > /dev/null 2>&1;
 then
```

```
            local error_message=$(echo "$response" | jq -r
 '.Error.Message // .Error.Code // .error // "Unknown error"' 2>/dev/null)
            handle_error "$operation failed: $error_message"
        fi
    else
        # Fallback to grep-based detection with optimized pattern
        if echo "$response" | grep -qE '"[Ee]rror":|"error":'; then
            handle_error "$operation failed: $response"
        fi
    fi
}

# Function to extract JSON field efficiently
extract_json_field() {
    local json="$1"
    local field="$2"

    if [[ "$USE_JQ" == "true" ]]; then
        echo "$json" | jq -r "$field" 2>/dev/null
    else
        echo "$json" | grep -oP "\"${field}\":\s*\"\K[^\"]+|\"${field}\":\s*
\K[^,}]+" | head -1
    fi
}

# Function to securely wait for resource propagation with exponential backoff
wait_for_propagation() {
    local resource_type="$1"
    local wait_time="${2:-10}"

    # Validate wait_time is a positive integer
    if ! [[ "$wait_time" =~ ^[0-9]+$ ]] || [ "$wait_time" -lt 1 ] ||
 [ "$wait_time" -gt 300 ]; then
        echo "WARNING: Invalid wait time $wait_time, using default 10 seconds"
        wait_time=10
    fi

    echo "Waiting for $resource_type to propagate ($wait_time seconds)..."
    sleep "$wait_time"
}

# Function to validate JSON file efficiently
validate_json_file() {
    local file="$1"
```

```
    if [[ "$USE_JQ" == "true" ]]; then
        if ! jq empty "$file" 2>/dev/null; then
            handle_error "Invalid JSON in $file"
        fi
    fi
}

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created:"
    if [ -n "${STATE_MACHINE_ARN:-}" ]; then
        echo "- State Machine: $STATE_MACHINE_ARN"
    fi
    if [ -n "${ROLE_NAME:-}" ]; then
        echo "- IAM Role: $ROLE_NAME"
    fi
    if [ -n "${POLICY_ARN:-}" ]; then
        echo "- IAM Policy: $POLICY_ARN"
    fi
    if [ -n "${STEPFUNCTIONS_POLICY_ARN:-}" ]; then
        echo "- Step Functions Policy: $STEPFUNCTIONS_POLICY_ARN"
    fi

    echo "Attempting to clean up resources..."
    cleanup
    exit 1
}

# Function to securely clean up resources with parallel deletion
cleanup() {
    echo "Cleaning up resources..."

    # Delete state machine if it exists
    if [ -n "${STATE_MACHINE_ARN:-}" ]; then
        echo "Deleting state machine: $STATE_MACHINE_ARN"
        aws stepfunctions delete-state-machine --state-machine-arn
 "$STATE_MACHINE_ARN" 2>/dev/null &
    fi

    # Detach and delete policies if they exist
    if [ -n "${POLICY_ARN:-}" ] && [ -n "${ROLE_NAME:-}" ]; then
        echo "Detaching Comprehend policy $POLICY_ARN from role $ROLE_NAME"
```

```
        aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$POLICY_ARN" 2>/dev/null &
    fi

    if [ -n "${STEPFUNCTIONS_POLICY_ARN:-}" ] && [ -n "${ROLE_NAME:-}" ]; then
        echo "Detaching Step Functions policy $STEPFUNCTIONS_POLICY_ARN from role
 $ROLE_NAME"
        aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$STEPFUNCTIONS_POLICY_ARN" 2>/dev/null &
    fi

    # Wait for detach operations to complete
    wait 2>/dev/null || true

    # Delete custom policies if they exist
    if [ -n "${POLICY_ARN:-}" ]; then
        echo "Deleting Comprehend policy: $POLICY_ARN"
        aws iam delete-policy --policy-arn "$POLICY_ARN" 2>/dev/null &
    fi

    if [ -n "${STEPFUNCTIONS_POLICY_ARN:-}" ]; then
        echo "Deleting Step Functions policy: $STEPFUNCTIONS_POLICY_ARN"
        aws iam delete-policy --policy-arn "$STEPFUNCTIONS_POLICY_ARN" 2>/dev/
null &
    fi

    # Wait for policy deletion to complete
    wait 2>/dev/null || true

    # Delete role if it exists
    if [ -n "${ROLE_NAME:-}" ]; then
        echo "Deleting role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || echo "Failed
 to delete role"
    fi

    # Remove temporary files securely
    echo "Removing temporary files"
    local temp_files=(
        "hello-world.json"
        "updated-hello-world.json"
        "sentiment-hello-world.json"
        "step-functions-trust-policy.json"
        "comprehend-policy.json"
```

```
        "stepfunctions-policy.json"
        "input.json"
        "sentiment-input.json"
    )

    for file in "${temp_files[@]}"; do
        if [ -f "$file" ]; then
            if command -v shred &> /dev/null; then
                shred -vfz -n 3 "$file" 2>/dev/null || rm -f "$file"
            else
                rm -f "$file"
            fi
        fi
    done
}

# Security: Set trap to cleanup on script exit
trap cleanup EXIT

# Generate a secure random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
ROLE_NAME="StepFunctionsHelloWorldRole-${RANDOM_ID}"
POLICY_NAME="DetectSentimentPolicy-${RANDOM_ID}"
STATE_MACHINE_NAME="MyFirstStateMachine-${RANDOM_ID}"

echo "Using random identifier: $RANDOM_ID"
echo "Role name: $ROLE_NAME"
echo "Policy name: $POLICY_NAME"
echo "State machine name: $STATE_MACHINE_NAME"

# Step 1: Create the state machine definition
echo "Creating state machine definition..."
cat > hello-world.json << 'EOF'
{
  "Comment": "A Hello World example of the Amazon States Language using a Pass
 state",
  "StartAt": "SetVariables",
  "States": {
    "SetVariables": {
      "Type": "Pass",
      "Result": {
        "IsHelloWorldExample": true,
        "ExecutionWaitTimeInSeconds": 10
      },
```

```
      "Next": "IsHelloWorldExample"
    },
    "IsHelloWorldExample": {
      "Type": "Choice",
      "Choices": [
        {
          "Variable": "$.IsHelloWorldExample",
          "BooleanEquals": true,
          "Next": "WaitState"
        }
      ],
      "Default": "FailState"
    },
    "WaitState": {
      "Type": "Wait",
      "SecondsPath": "$.ExecutionWaitTimeInSeconds",
      "Next": "ParallelProcessing"
    },
    "ParallelProcessing": {
      "Type": "Parallel",
      "Branches": [
        {
          "StartAt": "Process1",
          "States": {
            "Process1": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 1"
              },
              "End": true
            }
          }
        },
        {
          "StartAt": "Process2",
          "States": {
            "Process2": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 2"
              },
              "End": true
            }
          }
```

```
        }
      ],
      "Next": "CheckpointState"
    },
    "CheckpointState": {
      "Type": "Pass",
      "Result": {
        "CheckpointMessage": "Workflow completed successfully!"
      },
      "Next": "SuccessState"
    },
    "SuccessState": {
      "Type": "Succeed"
    },
    "FailState": {
      "Type": "Fail",
      "Error": "NotHelloWorldExample",
      "Cause": "The IsHelloWorldExample value was false"
    }
  }
}
EOF

# Create IAM role trust policy
echo "Creating IAM role trust policy..."
cat > step-functions-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "states.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role
echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
```

```
  --assume-role-policy-document file://step-functions-trust-policy.json 2>&1)

check_api_error "$ROLE_RESULT" "Create IAM role"
echo "Role created successfully"
aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-step-functions-gs

# Get the role ARN
ROLE_ARN=$(extract_json_field "$ROLE_RESULT" ".Role.Arn")

if [ -z "$ROLE_ARN" ]; then
    handle_error "Failed to extract role ARN"
fi
echo "Role ARN: $ROLE_ARN"

# Create a custom policy for Step Functions with least privilege
echo "Creating custom policy for Step Functions..."
cat > stepfunctions-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "states:StartExecution",
        "states:DescribeExecution",
        "states:StopExecution"
      ],
      "Resource": "arn:aws:states:*:*:stateMachine:*"
    }
  ]
}
EOF

# Create the policy
echo "Creating Step Functions policy..."
STEPFUNCTIONS_POLICY_RESULT=$(aws iam create-policy \
  --policy-name "StepFunctionsPolicy-${RANDOM_ID}" \
  --policy-document file://stepfunctions-policy.json 2>&1)

check_api_error "$STEPFUNCTIONS_POLICY_RESULT" "Create Step Functions policy"
echo "Step Functions policy created successfully"

# Get the policy ARN
```

```
STEPFUNCTIONS_POLICY_ARN=$(extract_json_field "$STEPFUNCTIONS_POLICY_RESULT"
 ".Policy.Arn")

if [ -z "$STEPFUNCTIONS_POLICY_ARN" ]; then
    handle_error "Failed to extract Step Functions policy ARN"
fi
echo "Step Functions policy ARN: $STEPFUNCTIONS_POLICY_ARN"

# Attach policy to the role
echo "Attaching Step Functions policy to role..."
ATTACH_RESULT=$(aws iam attach-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-arn "$STEPFUNCTIONS_POLICY_ARN" 2>&1)

if [ $? -ne 0 ]; then
    handle_error "Failed to attach Step Functions policy to role: $ATTACH_RESULT"
fi

# Wait for role to propagate (IAM changes can take time to propagate)
wait_for_propagation "IAM role" 8

# Create state machine
echo "Creating state machine: $STATE_MACHINE_NAME"
SM_RESULT=$(aws stepfunctions create-state-machine \
  --name "$STATE_MACHINE_NAME" \
  --definition file://hello-world.json \
  --role-arn "$ROLE_ARN" \
  --type STANDARD \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-step-functions-gs
 2>&1)

check_api_error "$SM_RESULT" "Create state machine"
echo "State machine created successfully"

# Get the state machine ARN
STATE_MACHINE_ARN=$(extract_json_field "$SM_RESULT" ".stateMachineArn")

if [ -z "$STATE_MACHINE_ARN" ]; then
    handle_error "Failed to extract state machine ARN"
fi
echo "State machine ARN: $STATE_MACHINE_ARN"

# Step 2: Start the state machine execution
echo "Starting state machine execution..."
```

```
EXEC_RESULT=$(aws stepfunctions start-execution \
  --state-machine-arn "$STATE_MACHINE_ARN" \
  --name "hello001-${RANDOM_ID}" 2>&1)

check_api_error "$EXEC_RESULT" "Start execution"
echo "Execution started successfully"

# Get the execution ARN
EXECUTION_ARN=$(extract_json_field "$EXEC_RESULT" ".executionArn")

if [ -z "$EXECUTION_ARN" ]; then
    handle_error "Failed to extract execution ARN"
fi
echo "Execution ARN: $EXECUTION_ARN"

# Wait for execution to complete (the workflow has a 10-second wait state)
echo "Waiting for execution to complete (12 seconds)..."
sleep 12

# Check execution status
echo "Checking execution status..."
EXEC_STATUS=$(aws stepfunctions describe-execution \
  --execution-arn "$EXECUTION_ARN" 2>&1)

echo "Execution status: $EXEC_STATUS"

# Step 3: Update state machine to process external input
echo "Updating state machine to process external input..."
cat > updated-hello-world.json << 'EOF'
{
  "Comment": "A Hello World example of the Amazon States Language using a Pass
 state",
  "StartAt": "SetVariables",
  "States": {
    "SetVariables": {
      "Type": "Pass",
      "Parameters": {
        "IsHelloWorldExample.$": "$.hello_world",
        "ExecutionWaitTimeInSeconds.$": "$.wait"
      },
      "Next": "IsHelloWorldExample"
    },
    "IsHelloWorldExample": {
      "Type": "Choice",
```

```
      "Choices": [
        {
          "Variable": "$.IsHelloWorldExample",
          "BooleanEquals": true,
          "Next": "WaitState"
        }
      ],
      "Default": "FailState"
    },
    "WaitState": {
      "Type": "Wait",
      "SecondsPath": "$.ExecutionWaitTimeInSeconds",
      "Next": "ParallelProcessing"
    },
    "ParallelProcessing": {
      "Type": "Parallel",
      "Branches": [
        {
          "StartAt": "Process1",
          "States": {
            "Process1": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 1"
              },
              "End": true
            }
          }
        },
        {
          "StartAt": "Process2",
          "States": {
            "Process2": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 2"
              },
              "End": true
            }
          }
        }
      ],
      "Next": "CheckpointState"
    },
```

```
    "CheckpointState": {
      "Type": "Pass",
      "Result": {
        "CheckpointMessage": "Workflow completed successfully!"
      },
      "Next": "SuccessState"
    },
    "SuccessState": {
      "Type": "Succeed"
    },
    "FailState": {
      "Type": "Fail",
      "Error": "NotHelloWorldExample",
      "Cause": "The IsHelloWorldExample value was false"
    }
  }
}
EOF

# Update state machine
echo "Updating state machine..."
UPDATE_RESULT=$(aws stepfunctions update-state-machine \
  --state-machine-arn "$STATE_MACHINE_ARN" \
  --definition file://updated-hello-world.json \
  --role-arn "$ROLE_ARN" 2>&1)

check_api_error "$UPDATE_RESULT" "Update state machine"
echo "State machine updated successfully"

# Create input file with strict validation
echo "Creating input file..."
cat > input.json << 'EOF'
{
  "wait": 5,
  "hello_world": true
}
EOF

# Validate input JSON
validate_json_file "input.json"

# Start execution with input
echo "Starting execution with input..."
EXEC2_RESULT=$(aws stepfunctions start-execution \
```

```
  --state-machine-arn "$STATE_MACHINE_ARN" \
  --name "hello002-${RANDOM_ID}" \
  --input file://input.json 2>&1)

check_api_error "$EXEC2_RESULT" "Start execution with input"
echo "Execution with input started successfully"

# Get the execution ARN
EXECUTION2_ARN=$(extract_json_field "$EXEC2_RESULT" ".executionArn")

if [ -z "$EXECUTION2_ARN" ]; then
    handle_error "Failed to extract execution ARN"
fi
echo "Execution ARN: $EXECUTION2_ARN"

# Wait for execution to complete (the workflow has a 5-second wait state)
echo "Waiting for execution to complete (8 seconds)..."
sleep 8

# Check execution status
echo "Checking execution status..."
EXEC2_STATUS=$(aws stepfunctions describe-execution \
  --execution-arn "$EXECUTION2_ARN" 2>&1)

echo "Execution status: $EXEC2_STATUS"

# Step 4: Integrate Amazon Comprehend for sentiment analysis (if available)
if [[ "$SKIP_COMPREHEND" == "false" ]]; then
    echo "Creating policy for Amazon Comprehend access with least privilege..."
    cat > comprehend-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "comprehend:DetectSentiment"
      ],
      "Resource": "*"
    }
  ]
}
EOF
```

```
    # Create policy
    echo "Creating IAM policy: $POLICY_NAME"
    POLICY_RESULT=$(aws iam create-policy \
      --policy-name "$POLICY_NAME" \
      --policy-document file://comprehend-policy.json 2>&1)

    check_api_error "$POLICY_RESULT" "Create Comprehend policy"
    echo "Comprehend policy created successfully"

    # Get policy ARN
    POLICY_ARN=$(extract_json_field "$POLICY_RESULT" ".Policy.Arn")

    if [ -z "$POLICY_ARN" ]; then
        handle_error "Failed to extract policy ARN"
    fi
    echo "Policy ARN: $POLICY_ARN"

    # Attach policy to role
    echo "Attaching policy to role..."
    ATTACH2_RESULT=$(aws iam attach-role-policy \
      --role-name "$ROLE_NAME" \
      --policy-arn "$POLICY_ARN" 2>&1)

    if [ $? -ne 0 ]; then
        handle_error "Failed to attach policy to role: $ATTACH2_RESULT"
    fi

    # Create updated state machine definition with sentiment analysis
    echo "Creating updated state machine definition with sentiment analysis..."
    cat > sentiment-hello-world.json << 'EOF'
{
  "Comment": "A Hello World example with sentiment analysis",
  "StartAt": "SetVariables",
  "States": {
    "SetVariables": {
      "Type": "Pass",
      "Parameters": {
        "IsHelloWorldExample.$": "$.hello_world",
        "ExecutionWaitTimeInSeconds.$": "$.wait",
        "FeedbackComment.$": "$.feedback_comment"
      },
      "Next": "IsHelloWorldExample"
    },
    "IsHelloWorldExample": {
```

```
      "Type": "Choice",
      "Choices": [
        {
          "Variable": "$.IsHelloWorldExample",
          "BooleanEquals": true,
          "Next": "WaitState"
        }
      ],
      "Default": "DetectSentiment"
    },
    "WaitState": {
      "Type": "Wait",
      "SecondsPath": "$.ExecutionWaitTimeInSeconds",
      "Next": "ParallelProcessing"
    },
    "ParallelProcessing": {
      "Type": "Parallel",
      "Branches": [
        {
          "StartAt": "Process1",
          "States": {
            "Process1": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 1"
              },
              "End": true
            }
          }
        },
        {
          "StartAt": "Process2",
          "States": {
            "Process2": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 2"
              },
              "End": true
            }
          }
        }
      ],
      "Next": "CheckpointState"
```

```
    },
    "CheckpointState": {
      "Type": "Pass",
      "Result": {
        "CheckpointMessage": "Workflow completed successfully!"
      },
      "Next": "SuccessState"
    },
    "DetectSentiment": {
      "Type": "Task",
      "Resource": "arn:aws:states:::aws-sdk:comprehend:detectSentiment",
      "Parameters": {
        "LanguageCode": "en",
        "Text.$": "$.FeedbackComment"
      },
      "Next": "SuccessState"
    },
    "SuccessState": {
      "Type": "Succeed"
    }
  }
}
EOF

    # Validate sentiment state machine JSON
    validate_json_file "sentiment-hello-world.json"

    # Wait for IAM changes to propagate
    wait_for_propagation "IAM changes" 8

    # Update state machine
    echo "Updating state machine with sentiment analysis..."
    UPDATE2_RESULT=$(aws stepfunctions update-state-machine \
      --state-machine-arn "$STATE_MACHINE_ARN" \
      --definition file://sentiment-hello-world.json \
      --role-arn "$ROLE_ARN" 2>&1)

    check_api_error "$UPDATE2_RESULT" "Update state machine with sentiment
 analysis"
    echo "State machine updated with sentiment analysis successfully"

    # Create input file with feedback comment
    echo "Creating input file with feedback comment..."
    cat > sentiment-input.json << 'EOF'
```

```
{
  "hello_world": false,
  "wait": 5,
  "feedback_comment": "This getting started with Step Functions workshop is a
 challenge!"
}
EOF

    # Validate sentiment input JSON
    validate_json_file "sentiment-input.json"

    # Start execution with sentiment analysis input
    echo "Starting execution with sentiment analysis input..."
    EXEC3_RESULT=$(aws stepfunctions start-execution \
      --state-machine-arn "$STATE_MACHINE_ARN" \
      --name "hello003-${RANDOM_ID}" \
      --input file://sentiment-input.json 2>&1)

    check_api_error "$EXEC3_RESULT" "Start execution with sentiment analysis"
    echo "Execution with sentiment analysis started successfully"

    # Get the execution ARN
    EXECUTION3_ARN=$(extract_json_field "$EXEC3_RESULT" ".executionArn")

    if [ -z "$EXECUTION3_ARN" ]; then
        handle_error "Failed to extract execution ARN"
    fi
    echo "Execution ARN: $EXECUTION3_ARN"

    # Wait for execution to complete
    echo "Waiting for execution to complete (3 seconds)..."
    sleep 3

    # Check execution status
    echo "Checking execution status..."
    EXEC3_STATUS=$(aws stepfunctions describe-execution \
      --execution-arn "$EXECUTION3_ARN" 2>&1)

    echo "Execution status: $EXEC3_STATUS"
else
    echo "Skipping Amazon Comprehend integration (not available in
 $CURRENT_REGION)"
    EXECUTION3_ARN=""
fi
```

```
# Display summary of resources created
echo ""
echo "=========================================="
echo "RESOURCES CREATED"
echo "=========================================="
echo "State Machine: $STATE_MACHINE_ARN"
echo "IAM Role: $ROLE_NAME"
echo "Step Functions Policy: StepFunctionsPolicy-${RANDOM_ID}
 ($STEPFUNCTIONS_POLICY_ARN)"
if [[ "$SKIP_COMPREHEND" == "false" ]]; then
    echo "Comprehend Policy: $POLICY_NAME ($POLICY_ARN)"
fi
echo "Executions:"
echo "  - hello001-${RANDOM_ID}: $EXECUTION_ARN"
echo "  - hello002-${RANDOM_ID}: $EXECUTION2_ARN"
if [[ "$SKIP_COMPREHEND" == "false" ]]; then
    echo "  - hello003-${RANDOM_ID}: $EXECUTION3_ARN"
fi
echo "=========================================="

# Cleanup
echo ""
echo "=========================================="
echo "CLEANUP"
echo "=========================================="
echo "Auto-cleanup enabled. Cleaning up resources..."

echo "All resources have been cleaned up."

echo "Script completed successfully!"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreatePolicy
  - CreateRole
  - CreateStateMachine
  - DeletePolicy
  - DeleteRole

- DeleteStateMachine
- DescribeExecution
- DetachRolePolicy
- StartExecution
- UpdateStateMachine

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Gérez les clés d'accès IAM à l'aide d'un AWS Kit SDK

L'exemple de code suivant montre comment gérer les clés d'accès.

> **Warning**
>
> Afin d'éviter les risques de sécurité, n'employez pas les utilisateurs IAM pour l'authentification lorsque vous développez des logiciels spécialisés ou lorsque vous travaillez avec des données réelles. Préférez la fédération avec un fournisseur d'identité tel que AWS IAM Identity Center.

- Créez et répertoriez les clés d'accès.
- Découvrez quand et comment une clé d'accès a été utilisée pour la dernière fois.
- Mettez à jour et supprimez les clés d'accès.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de clé d'accès IAM.

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

iam = boto3.resource("iam")

def list_keys(user_name):
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
    else:
        return keys



def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
```

```
        raise
    else:
        return key_pair



def get_last_use(key_id):
    """
    Gets information about when and how a key was last used.

    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
    try:
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
            "Key %s was last used by %s on %s to access %s.",
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
    except ClientError:
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
        return response



def update_key(user_name, key_id, activate):
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
    :param activate: When True, the key is activated. Otherwise, the key is
 deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
```

```
        if activate:
            key.activate()
        else:
            key.deactivate()
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
 key_id)
    except ClientError:
        logger.exception(
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
 key_id
        )
        raise



def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

Utilisez les fonctions d'enveloppe pour effectuer des actions de clé d'accès pour l'utilisateur actuel.

```
def usage_demo():
    """Shows how to create and manage access keys."""

    def print_keys():
        """Gets and prints the current keys for a user."""
        current_keys = list_keys(current_user_name)
```

```
        print("The current user's keys are now:")
        print(*[f"{key.id}: {key.status}" for key in current_keys], sep="\n")

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management access key demo.")
    print("-" * 88)
    current_user_name = iam.CurrentUser().user_name
    print(
        f"This demo creates an access key for the current user "
        f"({current_user_name}), manipulates the key in a few ways, and then "
        f"deletes it."
    )
    all_keys = list_keys(current_user_name)
    if len(all_keys) == 2:
        print(
            "The current user already has the maximum of 2 access keys. To run "
            "this demo, either delete one of the access keys or use a user "
            "that has only 1 access key."
        )
    else:
        new_key = create_key(current_user_name)
        print(f"Created a new key with id {new_key.id} and secret
{new_key.secret}.")
        print_keys()
        existing_key = next(key for key in all_keys if key != new_key)
        last_use = get_last_use(existing_key.id)["AccessKeyLastUsed"]
        print(
            f"Key {all_keys[0].id} was last used to access
{last_use['ServiceName']} "
            f"on {last_use['LastUsedDate']}"
        )
        update_key(current_user_name, new_key.id, False)
        print(f"Key {new_key.id} is now deactivated.")
        print_keys()
        delete_key(current_user_name, new_key.id)
        print_keys()
        print("Thanks for watching!")
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK for Python (Boto3).
  - CreateAccessKey
  - DeleteAccessKey
  - GetAccessKeyLastUsed
  - ListAccessKeys
  - UpdateAccessKey

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Gérez les politiques IAM à l'aide d'un AWS Kit SDK

L'exemple de code suivant illustre comment :

- Créez et répertoriez des stratégies.
- Créez et obtenez des versions de stratégies.
- Rétablissez une stratégie à une version précédente.
- Supprimez des stratégies.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de politique IAM.

```
import json
import logging
import operator
import pprint
```

```
import time

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::amzn-s3-demo-bucket/*' to allow actions on
 all objects
                         in the bucket named 'amzn-s3-demo-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy
```

```
def list_policies(scope):
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
                  'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
        return policies



def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
```

```
            "Created policy version %s for policy %s.",
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
        logger.exception("Couldn't create a policy version for %s.", policy_arn)
        raise
    else:
        return policy_version



def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement



def rollback_policy_version(policy_arn):
    """
    Rolls back to the previous default policy, if it exists.

    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
    4. Deletes the old default version.
```

```
    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
        raise

    default_version = None
    rollback_version = None
    try:
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
        rollback_version.set_as_default()
        logger.info("Set %s as the default version.",
rollback_version.version_id)
        default_version.delete()
        logger.info("Deleted original default version %s.",
default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
so "
                "nothing to roll back to.",
                default_version.version_id,
                policy_arn,
            )
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
        return rollback_version
```

```
def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

Utilisez les fonctions d'enveloppe pour créer des politiques, mettre à jour des versions et obtenir des informations à leur sujet.

```
def usage_demo():
    """Shows how to use the policy functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management policy demo.")
    print("-" * 88)
    print(
        "Policies let you define sets of permissions that can be attached to "
        "other IAM resources, like users and roles."
    )
    bucket_arn = f"arn:aws:s3:::amzn-s3-demo-bucket"
    policy = create_policy(
        "demo-iam-policy",
        "Policy for IAM demonstration.",
        ["s3:ListObjects"],
        bucket_arn,
    )
    print(f"Created policy {policy.policy_name}.")
    policies = list_policies("Local")
    print(f"Your account has {len(policies)} managed policies:")
    print(*[pol.policy_name for pol in policies], sep=", ")
    time.sleep(1)
```

```
    policy_version = create_policy_version(
        policy.arn, ["s3:PutObject"], bucket_arn, True
    )
    print(
        f"Added policy version {policy_version.version_id} to policy "
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
    print(f"The default policy statement for {policy.policy_name} is:")
    pprint.pprint(default_statement)
    rollback_version = rollback_policy_version(policy.arn)
    print(
        f"Rolled back to version {rollback_version.version_id} for "
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
    print(f"The default policy statement for {policy.policy_name} is now:")
    pprint.pprint(default_statement)
    delete_policy(policy.arn)
    print(f"Deleted policy {policy.policy_name}.")
    print("Thanks for watching!")
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK for Python (Boto3).
  - CreatePolicy
  - CreatePolicyVersion
  - DeletePolicy
  - DeletePolicyVersion
  - GetPolicyVersion
  - ListPolicies
  - ListPolicyVersions
  - SetDefaultPolicyVersion

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Gérez les rôles IAM à l'aide d'un AWS Kit SDK

L'exemple de code suivant illustre comment :

- Créez un rôle IAM.
- Attacher et détacher des stratégies à/d'un rôle
- Supprimer un rôle

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de rôle IAM.

```
import json
import logging
import pprint

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
    :return: The newly created role.
    """
    trust_policy = {
        "Version":"2012-10-17",
        "Statement": [
```

```
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
            }
            for service in allowed_services
        ],
    }

    try:
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
    else:
        return role



def attach_policy(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise



def detach_policy(role_name, policy_arn):
    """
    Detaches a policy from a role.
```

```
    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise



def delete_role(role_name):
    """
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

Utilisez les fonctions d'enveloppe pour créer un rôle, puis attacher et détacher une politique.

```
def usage_demo():
    """Shows how to use the role functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management role demo.")
    print("-" * 88)
    print(
        "Roles let you define sets of permissions and can be assumed by "
        "other entities, like users and services."
    )
```

```
    print("The first 10 roles currently in your account are:")
    roles = list_roles(10)
    print(f"The inline policies for role {roles[0].name} are:")
    list_policies(roles[0].name)
    role = create_role(
        "demo-iam-role", ["lambda.amazonaws.com",
"batchoperations.s3.amazonaws.com"]
    )
    print(f"Created role {role.name}, with trust policy:")
    pprint.pprint(role.assume_role_policy_document)
    policy_arn = "arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess"
    attach_policy(role.name, policy_arn)
    print(f"Attached policy {policy_arn} to {role.name}.")
    print(f"Policies attached to role {role.name} are:")
    list_attached_policies(role.name)
    detach_policy(role.name, policy_arn)
    print(f"Detached policy {policy_arn} from {role.name}.")
    delete_role(role.name)
    print(f"Deleted {role.name}.")
    print("Thanks for watching!")
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK for Python (Boto3).
  - AttachRolePolicy
  - CreateRole
  - DeleteRole
  - DetachRolePolicy

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Gérez votre compte IAM à l'aide d'un AWS Kit SDK

L'exemple de code suivant illustre comment :

- Obtenez et mettez à jour l'alias du compte.

- Générez un rapport sur les utilisateurs et les informations d'identification.
- Obtenez un résumé de l'utilisation du compte.
- Obtenez des informations pour tous les utilisateurs, les groupes, les rôles et les politiques dans votre compte, y compris les relations entre eux.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions du compte IAM.

```
import logging
import pprint
import sys
import time
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def list_aliases():
    """
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
        else:
            logger.info("Got no aliases for your account.")
```

```
    except ClientError:
        logger.exception("Couldn't list aliases for your account.")
        raise
    else:
        return response["AccountAliases"]



def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
        raise



def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise



def generate_credential_report():
    """
```

```
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
    to get the latest report. A new report can be generated a minimum of four
 hours
    after the last one was generated.
    """
    try:
        response = iam.meta.client.generate_credential_report()
        logger.info(
            "Generating credentials report for your account. " "Current state is
 %s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
 account.")
        raise
    else:
        return response



def get_credential_report():
    """
    Gets the most recently generated credentials report about the current
 account.

    :return: The credentials report.
    """
    try:
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]



def get_summary():
    """
    Gets a summary of account usage.
```

```
    :return: The summary of account usage.
    """
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
        raise
    else:
        return summary.summary_map



def get_authorization_details(response_filter):
    """
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
 such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

Appelez les fonctions d'encapsulation pour modifier l'alias du compte et obtenir des rapports sur le compte.

```
def usage_demo():
    """Shows how to use the account functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
```

```
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management account demo.")
    print("-" * 88)
    print(
        "Setting an account alias lets you use the alias in your sign-in URL "
        "instead of your account number."
    )
    old_aliases = list_aliases()
    if len(old_aliases) > 0:
        print(f"Your account currently uses '{old_aliases[0]}' as its alias.")
    else:
        print("Your account currently has no alias.")
    for index in range(1, 3):
        new_alias = f"alias-{index}-{time.time_ns()}"
        print(f"Setting your account alias to {new_alias}")
        create_alias(new_alias)
    current_aliases = list_aliases()
    print(f"Your account alias is now {current_aliases}.")
    delete_alias(current_aliases[0])
    print(f"Your account now has no alias.")
    if len(old_aliases) > 0:
        print(f"Restoring your original alias back to {old_aliases[0]}...")
        create_alias(old_aliases[0])

    print("-" * 88)
    print("You can get various reports about your account.")
    print("Let's generate a credentials report...")
    report_state = None
    while report_state != "COMPLETE":
        cred_report_response = generate_credential_report()
        old_report_state = report_state
        report_state = cred_report_response["State"]
        if report_state != old_report_state:
            print(report_state, sep="")
        else:
            print(".", sep="")
        sys.stdout.flush()
        time.sleep(1)
    print()
    cred_report = get_credential_report()
    col_count = 3
    print(f"Got credentials report. Showing only the first {col_count} columns.")
    cred_lines = [
```

```
        line.split(",")[:col_count] for line in
 cred_report.decode("utf-8").split("\n")
    ]
    col_width = max([len(item) for line in cred_lines for item in line]) + 2
    for line in cred_report.decode("utf-8").split("\n"):
        print(
            "".join(element.ljust(col_width) for element in line.split(",")
[:col_count])
        )

    print("-" * 88)
    print("Let's get an account summary.")
    summary = get_summary()
    print("Here's your summary:")
    pprint.pprint(summary)

    print("-" * 88)
    print("Let's get authorization details!")
    details = get_authorization_details([])
    see_details = input("These are pretty long, do you want to see them (y/n)? ")
    if see_details.lower() == "y":
        pprint.pprint(details)

    print("-" * 88)
    pw_policy_created = None
    see_pw_policy = input("Want to see the password policy for the account (y/n)?
 ")
    if see_pw_policy.lower() == "y":
        while True:
            if print_password_policy():
                break
            else:
                answer = input(
                    "Do you want to create a default password policy (y/n)? "
                )
                if answer.lower() == "y":
                    pw_policy_created = iam.create_account_password_policy()
                else:
                    break
    if pw_policy_created is not None:
        answer = input("Do you want to delete the password policy (y/n)? ")
        if answer.lower() == "y":
            pw_policy_created.delete()
            print("Password policy deleted.")
```

```
    print("The SAML providers for your account are:")
    list_saml_providers(10)

    print("-" * 88)
    print("Thanks for watching.")
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK for Python (Boto3).
  - CreateAccountAlias
  - DeleteAccountAlias
  - GenerateCredentialReport
  - GetAccountAuthorizationDetails
  - GetAccountSummary
  - GetCredentialReport
  - ListAccountAliases

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Transférer les secrets codés en dur vers un stockage secret sécurisé

L'exemple de code suivant illustre comment :

- Création de rôles IAM
- Création d'un secret dans Secrets Manager
- Mettez à jour le code de votre application
- Mettre à jour le secret
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Script to move hardcoded secrets to AWS Secrets Manager
# This script demonstrates how to create IAM roles, store a secret in AWS Secrets
 Manager,
# and set up appropriate permissions

set -euo pipefail

# Set up logging
LOG_FILE="secrets_manager_tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS Secrets Manager tutorial script at $(date)"
echo "============================================================"

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -qi "error"; then
        echo "ERROR: Command failed: $cmd"
        echo "$output"
        cleanup_resources
        exit 1
    fi
}

# Function to generate a random identifier using secure method
generate_random_id() {
    python3 -c "import secrets; print('sm' + secrets.token_hex(4))"
```

```
}

# Function to safely clean up resources
cleanup_resources() {
    echo ""
    echo "========================================="
    echo "RESOURCES CREATED"
    echo "========================================="

    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Secret: $SECRET_NAME"
    fi

    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "IAM Role: $RUNTIME_ROLE_NAME"
    fi

    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "IAM Role: $ADMIN_ROLE_NAME"
    fi

    echo ""
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "Cleaning up all created resources..."

    # Delete secret if it exists
    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Deleting secret: $SECRET_NAME"
        aws secretsmanager delete-secret --secret-id "$SECRET_NAME" --force-
delete-without-recovery 2>/dev/null || true
    fi

    # Detach policies and delete runtime role if it exists
    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "Deleting inline policies from runtime role: $RUNTIME_ROLE_NAME"
        for policy in $(aws iam list-role-policies --role-name
 "$RUNTIME_ROLE_NAME" --query 'PolicyNames[]' --output text 2>/dev/null || true);
 do
            aws iam delete-role-policy --role-name "$RUNTIME_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
        done
        echo "Deleting IAM role: $RUNTIME_ROLE_NAME"
```

```
        aws iam delete-role --role-name "$RUNTIME_ROLE_NAME" 2>/dev/null || true
    fi

    # Detach policies and delete admin role if it exists
    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "Detaching policy from role: $ADMIN_ROLE_NAME"
        aws iam detach-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-arn
 "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>/dev/null || true

        for policy in $(aws iam list-role-policies --role-name "$ADMIN_ROLE_NAME"
 --query 'PolicyNames[]' --output text 2>/dev/null || true); do
            aws iam delete-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
        done

        echo "Deleting IAM role: $ADMIN_ROLE_NAME"
        aws iam delete-role --role-name "$ADMIN_ROLE_NAME" 2>/dev/null || true
    fi

    echo "Cleanup completed."
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Running cleanup..."; cleanup_resources' INT TERM
 EXIT

# Generate random identifiers for resources
ADMIN_ROLE_NAME="SecretsManagerAdmin-$(generate_random_id)"
RUNTIME_ROLE_NAME="RoleToRetrieveSecretAtRuntime-$(generate_random_id)"
SECRET_NAME="MyAPIKey-$(generate_random_id)"

echo "Using the following resource names:"
echo "Admin Role: $ADMIN_ROLE_NAME"
echo "Runtime Role: $RUNTIME_ROLE_NAME"
echo "Secret Name: $SECRET_NAME"
echo ""

# Step 1: Create IAM roles
echo "Creating IAM roles..."

# Create assume role policy document
ASSUME_ROLE_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF
)

# Create the SecretsManagerAdmin role
echo "Creating admin role: $ADMIN_ROLE_NAME"
ADMIN_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$ADMIN_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$ADMIN_ROLE_OUTPUT" "create-role for admin"
echo "$ADMIN_ROLE_OUTPUT"
aws iam tag-role --role-name "$ADMIN_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs

# Attach the SecretsManagerReadWrite policy to the admin role
echo "Attaching SecretsManagerReadWrite policy to admin role"
ATTACH_POLICY_OUTPUT=$(aws iam attach-role-policy \
    --role-name "$ADMIN_ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>&1)

check_error "$ATTACH_POLICY_OUTPUT" "attach-role-policy for admin"
echo "Policy attached successfully"

# Create the RoleToRetrieveSecretAtRuntime role
echo "Creating runtime role: $RUNTIME_ROLE_NAME"
RUNTIME_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$RUNTIME_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$RUNTIME_ROLE_OUTPUT" "create-role for runtime"
echo "$RUNTIME_ROLE_OUTPUT"
aws iam tag-role --role-name "$RUNTIME_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs
```

```
# Wait for roles to be fully created
echo "Waiting for IAM roles to be fully created..."
sleep 10

# Step 2: Create a secret in AWS Secrets Manager
echo "Creating secret in AWS Secrets Manager..."

# Generate secure secret value using environment variable or secure method
# WARNING: In production, use secure methods to inject secrets (AWS CodeBuild,
 parameter store, etc.)
if [ -z "${TUTORIAL_SECRET_VALUE:-}" ]; then
    SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_client_id','ClientSecret':__import__('secrets').token_u
else
    SECRET_VALUE="$TUTORIAL_SECRET_VALUE"
fi

CREATE_SECRET_OUTPUT=$(aws secretsmanager create-secret \
    --name "$SECRET_NAME" \
    --description "API key for my application" \
    --secret-string "$SECRET_VALUE" \
    --add-replica-regions 'Region=us-east-1' \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-secrets-manager-gs
 2>&1)

check_error "$CREATE_SECRET_OUTPUT" "create-secret"
echo "$CREATE_SECRET_OUTPUT"

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text 2>&1)
check_error "$ACCOUNT_ID" "get-caller-identity"
echo "Account ID: $ACCOUNT_ID"

# Get secret ARN for precise resource policy
echo "Getting secret ARN..."
SECRET_ARN=$(aws secretsmanager describe-secret \
    --secret-id "$SECRET_NAME" \
    --query 'ARN' \
    --output text 2>&1)
check_error "$SECRET_ARN" "describe-secret"

# Add resource policy to the secret with least privilege
echo "Adding resource policy to secret..."
```

```
RESOURCE_POLICY=$(cat <<EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRuntimeRoleReadSecret",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::$ACCOUNT_ID:role/$RUNTIME_ROLE_NAME"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "$SECRET_ARN",
            "Condition": {
                "StringEquals": {
                    "secretsmanager:VersionStage": "AWSCURRENT"
                }
            }
        }
    ]
}
EOF
)

PUT_POLICY_OUTPUT=$(aws secretsmanager put-resource-policy \
    --secret-id "$SECRET_NAME" \
    --resource-policy "$RESOURCE_POLICY" \
    --block-public-policy 2>&1)

check_error "$PUT_POLICY_OUTPUT" "put-resource-policy"
echo "Resource policy added successfully"

# Enable rotation policy recommendation
echo "Enabling secret metadata tags for rotation tracking..."
aws secretsmanager tag-resource \
    --secret-id "$SECRET_NAME" \
    --tags Key=Purpose,Value=Tutorial Key=AutoRotation,Value=Recommended 2>/dev/
null || true

# Step 3: Demonstrate retrieving the secret
echo "Retrieving the secret value (for demonstration purposes)..."
GET_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$GET_SECRET_OUTPUT" "get-secret-value"
```

```
echo "Secret retrieved successfully. Secret metadata:"
echo "$GET_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"

# Step 4: Update the secret with new values
echo "Updating the secret with new values..."
UPDATE_SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_new_client_id','ClientSecret':__import__('secrets').tok

UPDATE_SECRET_OUTPUT=$(aws secretsmanager update-secret \
    --secret-id "$SECRET_NAME" \
    --secret-string "$UPDATE_SECRET_VALUE" 2>&1)

check_error "$UPDATE_SECRET_OUTPUT" "update-secret"
echo "Secret updated successfully"

# Step 5: Verify the updated secret
echo "Verifying the updated secret..."
VERIFY_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$VERIFY_SECRET_OUTPUT" "get-secret-value for verification"
echo "Updated secret retrieved successfully. Secret metadata:"
echo "$VERIFY_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"

# Step 6: Display rotation recommendations
echo ""
echo "Rotation Configuration Recommendations:"
echo "======================================="
DESCRIBE_OUTPUT=$(aws secretsmanager describe-secret --secret-id "$SECRET_NAME"
 2>&1)
if echo "$DESCRIBE_OUTPUT" | grep -q "RotationRules"; then
    echo "Current rotation configuration:"
    echo "$DESCRIBE_OUTPUT" | jq '.RotationRules' 2>/dev/null || echo "Rotation
 rules available"
else
    echo "No automatic rotation configured. Consider enabling rotation with:"
    echo "aws secretsmanager rotate-secret --secret-id $SECRET_NAME --rotation-
lambda-arn arn:aws:lambda:REGION:ACCOUNT:function:FUNCTION_NAME --rotation-rules
 AutomaticallyAfterDays=30"
fi
```

```
echo ""
echo "============================================================"
echo "Tutorial completed successfully!"
echo ""
echo "Summary of what we did:"
echo "1. Created IAM roles for managing and retrieving secrets"
echo "2. Created a secret in AWS Secrets Manager with secure generation"
echo "3. Added a least-privilege resource policy to control access to the secret"
echo "4. Retrieved the secret value (simulating application access)"
echo "5. Updated the secret with cryptographically secure values"
echo "6. Verified the updated secret"
echo ""
echo "Security best practices applied:"
echo "- Used cryptographically secure random ID generation"
echo "- Applied least-privilege resource policies with version stages"
echo "- Tagged resources for rotation tracking"
echo "- Blocked public access to secrets"
echo "- Used ARN-specific permissions instead of wildcards"
echo ""
echo "Next steps you might want to consider:"
echo "- Enable automatic secret rotation with AWS Lambda"
echo "- Implement secret caching in your application"
echo "- Set up CloudTrail logging for secret access auditing"
echo "- Use AWS CodeGuru Reviewer to find hardcoded secrets in your code"
echo "- For multi-region applications, replicate your secrets across regions"
echo "- Configure VPC endpoints for private access to Secrets Manager"
echo ""

echo "Script completed at $(date)"
exit 0
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreateRole
  - CreateSecret
  - DeleteRole
  - DeleteSecret
  - DetachRolePolicy

- GetCallerIdentity
- GetSecretValue
- PutResourcePolicy
- UpdateSecret

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Permet le Optimiseur de calcul AWS Fonction d'automatisation pour appliquer les actions recommandées

L'exemple de code suivant montre comment cette politique basée sur les autorisations permet à la fonctionnalité d' Optimiseur de calcul AWS automatisation d'appliquer les actions recommandées

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "aco-automation.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Politique pour activer l'automatisation dans toute votre organisation

L'exemple de code suivant montre comment cette politique basée sur les autorisations active l'automatisation dans toute votre organisation

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation",
            "Condition": {"StringLike": {"iam:AWSServiceName": "aco-
automation.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PutRolePolicy",
                "iam:AttachRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:UpdateEnrollmentConfiguration",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:AssociateAccounts",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:DisassociateAccounts",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:ListAccounts",
            "Resource": "*"
        }
    ]
```

```
}
```

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Politique pour activer l'automatisation pour votre compte

L'exemple de code suivant montre comment cette politique basée sur les autorisations active l'automatisation pour votre compte

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation",
            "Condition": {"StringLike": {"iam:AWSServiceName": "aco-
automation.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PutRolePolicy",
                "iam:AttachRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:UpdateEnrollmentConfiguration",
            "Resource": "*"
        }
    ]
}
```

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Politique pour accorder un accès complet à l'automatisation de l'optimiseur de calcul au compte de gestion d'une organisation

L'exemple de code suivant montre comment cette politique basée sur les autorisations accorde un accès complet à l'automatisation de l'optimiseur de calcul au compte de gestion d'une organisation

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "aco-automation:*",
               "ec2:DescribeVolumes",
               "organizations:ListAccounts",
               "organizations:DescribeOrganization",
               "organizations:DescribeAccount",
               "organizations:EnableAWSServiceAccess",
               "organizations:ListDelegatedAdministrators",
               "organizations:RegisterDelegatedAdministrator",
               "organizations:DeregisterDelegatedAdministrator"
            ],
            "Resource": "*"
        }
    ]
}
```

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Politique visant à accorder un accès complet à Compute Optimizer Automation en mode autonome AWS comptes

L'exemple de code suivant montre comment cette politique basée sur les autorisations accorde un accès complet à Compute Optimizer Automation pour les comptes autonomes AWS

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "aco-automation:*",
            "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        }
    ]
}
```

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Politique pour accorder un accès en lecture seule à l'automatisation de l'optimiseur de calcul au compte de gestion d'une organisation

L'exemple de code suivant montre comment cette politique basée sur les autorisations accorde un accès en lecture seule à l'automatisation de l'optimiseur de calcul au compte de gestion d'une organisation

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
               "aco-automation:GetEnrollmentConfiguration",
               "aco-automation:GetAutomationEvent",
               "aco-automation:GetAutomationRule",
               "aco-automation:ListAccounts",
               "aco-automation:ListAutomationEvents",
               "aco-automation:ListAutomationEventSteps",
               "aco-automation:ListAutomationEventSummaries",
               "aco-automation:ListAutomationRules",
               "aco-automation:ListAutomationRulePreview",
               "aco-automation:ListAutomationRulePreviewSummaries",
               "aco-automation:ListRecommendedActions",
               "aco-automation:ListRecommendedActionSummaries",
               "aco-automation:ListTagsForResource",
               "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        }
    ]
}
```

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Politique visant à accorder un accès en lecture seule à Compute Optimizer Automation en mode autonome AWS comptes

L'exemple de code suivant montre comment cette politique basée sur les autorisations accorde un accès en lecture seule à Compute Optimizer Automation pour les comptes autonomes AWS

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "aco-automation:GetEnrollmentConfiguration",
```

```
                "aco-automation:GetAutomationEvent",
                "aco-automation:GetAutomationRule",
                "aco-automation:ListAutomationEvents",
                "aco-automation:ListAutomationEventSteps",
                "aco-automation:ListAutomationEventSummaries",
                "aco-automation:ListAutomationRules",
                "aco-automation:ListAutomationRulePreview",
                "aco-automation:ListAutomationRulePreviewSummaries",
                "aco-automation:ListRecommendedActions",
                "aco-automation:ListRecommendedActionSummaries",
                "aco-automation:ListTagsForResource",
                "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        }
    ]
}
```

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Politique pour accorder les autorisations de rôle lié à un service pour l'automatisation de l'optimisation du calcul

L'exemple de code suivant montre comment cette politique basée sur les autorisations accorde les autorisations de rôle lié à un service pour l'automatisation de l'optimisation du calcul

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation",
            "Condition": {"StringLike": {"iam:AWSServiceName": "aco-
automation.amazonaws.com"}}
        },
```

```
        {
            "Effect": "Allow",
            "Action": "iam:PutRolePolicy",
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation"
        }
    ]
}
```

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Restaurer une version d'une politique IAM à l'aide d'un AWS Kit SDK

L'exemple de code suivant illustre comment :

- Obtenez la liste des versions de stratégie par ordre chronologique.
- Trouver la version de stratégie par défaut.
- Définissez la version précédente de la stratégie comme version par défaut.
- Supprimez l'ancienne version par défaut.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def rollback_policy_version(policy_arn):
    """
    Rolls back to the previous default policy, if it exists.

    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
```

```
    4. Deletes the old default version.

    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
        raise

    default_version = None
    rollback_version = None
    try:
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
        rollback_version.set_as_default()
        logger.info("Set %s as the default version.",
rollback_version.version_id)
        default_version.delete()
        logger.info("Deleted original default version %s.",
default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
so "
                "nothing to roll back to.",
                default_version.version_id,
                policy_arn,
            )
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
```

```
        return rollback_version
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans AWS SDK for Python (Boto3) API Reference.
  - DeletePolicyVersion
  - ListPolicyVersions
  - SetDefaultPolicyVersion

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts

L'exemple de code suivant illustre comment :

- Création de rôles IAM
- Création d'une CloudWatch alarme
- Création d'un modèle d'expérience
- Exécutez l'expérience
- Vérifiez les résultats
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# AWS FIS CPU Stress Test Tutorial Script
# This script automates the steps in the AWS FIS CPU stress test tutorial

#    approach using epoch time calculations that work across all Linux
 distributions

# Set up logging
LOG_FILE="fis-tutorial-$(date +%Y%m%d-%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS FIS CPU Stress Test Tutorial Script"
echo "Logging to $LOG_FILE"
echo "============================================="

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -i "error" > /dev/null; then
        # Ignore specific expected errors
        if [[ "$cmd" == *"aws fis get-experiment"* ]] && [[ "$output" ==
 *"ConfigurationFailure"* ]]; then
            echo "Note: Experiment failed due to configuration issue. This is
 expected in some cases."
            return 0
        fi

        echo "ERROR: Command failed: $cmd"
        echo "Output: $output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Cleaning up resources..."

    if [ -n "$EXPERIMENT_ID" ]; then
        echo "Stopping experiment $EXPERIMENT_ID if running..."
```

```
        aws fis stop-experiment --id "$EXPERIMENT_ID" 2>/dev/null || true
    fi

    if [ -n "$TEMPLATE_ID" ]; then
        echo "Deleting experiment template $TEMPLATE_ID..."
        aws fis delete-experiment-template --id "$TEMPLATE_ID" || true
    fi

    if [ -n "$INSTANCE_ID" ]; then
        echo "Terminating EC2 instance $INSTANCE_ID..."
        aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || true
    fi

    if [ -n "$ALARM_NAME" ]; then
        echo "Deleting CloudWatch alarm $ALARM_NAME..."
        aws cloudwatch delete-alarms --alarm-names "$ALARM_NAME" || true
    fi

    if [ -n "$INSTANCE_PROFILE_NAME" ]; then
        echo "Removing role from instance profile..."
        aws iam remove-role-from-instance-profile --instance-profile-name
"$INSTANCE_PROFILE_NAME" --role-name "$EC2_ROLE_NAME" || true

        echo "Deleting instance profile..."
        aws iam delete-instance-profile --instance-profile-name
"$INSTANCE_PROFILE_NAME" || true
    fi

    if [ -n "$FIS_ROLE_NAME" ]; then
        echo "Deleting FIS role policy..."
        aws iam delete-role-policy --role-name "$FIS_ROLE_NAME" --policy-name
"$FIS_POLICY_NAME" || true

        echo "Deleting FIS role..."
        aws iam delete-role --role-name "$FIS_ROLE_NAME" || true
    fi

    if [ -n "$EC2_ROLE_NAME" ]; then
        echo "Detaching policy from EC2 role..."
        aws iam detach-role-policy --role-name "$EC2_ROLE_NAME" --policy-arn
"arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore" || true

        echo "Deleting EC2 role..."
        aws iam delete-role --role-name "$EC2_ROLE_NAME" || true
```

```
    fi

    echo "Cleanup completed."
}

# Generate unique identifiers for resources
TIMESTAMP=$(date +%Y%m%d%H%M%S)
FIS_ROLE_NAME="FISRole-${TIMESTAMP}"
FIS_POLICY_NAME="FISPolicy-${TIMESTAMP}"
EC2_ROLE_NAME="EC2SSMRole-${TIMESTAMP}"
INSTANCE_PROFILE_NAME="EC2SSMProfile-${TIMESTAMP}"
ALARM_NAME="FIS-CPU-Alarm-${TIMESTAMP}"

# Track created resources
CREATED_RESOURCES=()

echo "Step 1: Creating IAM role for AWS FIS"
# Create trust policy file for AWS FIS
cat > fis-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "fis.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role for FIS
echo "Creating IAM role $FIS_ROLE_NAME for AWS FIS..."
FIS_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$FIS_ROLE_NAME" \
  --assume-role-policy-document file://fis-trust-policy.json)
check_error "$FIS_ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$FIS_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $FIS_ROLE_NAME")

# Create policy document for SSM actions
```

```
cat > fis-ssm-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:SendCommand",
        "ssm:ListCommands",
        "ssm:ListCommandInvocations"
      ],
      "Resource": "*"
    }
  ]
}
EOF

# Attach policy to the role
echo "Attaching policy $FIS_POLICY_NAME to role $FIS_ROLE_NAME..."
FIS_POLICY_OUTPUT=$(aws iam put-role-policy \
  --role-name "$FIS_ROLE_NAME" \
  --policy-name "$FIS_POLICY_NAME" \
  --policy-document file://fis-ssm-policy.json)
check_error "$FIS_POLICY_OUTPUT" "aws iam put-role-policy"
CREATED_RESOURCES+=("IAM Policy: $FIS_POLICY_NAME attached to $FIS_ROLE_NAME")

echo "Step 2: Creating IAM role for EC2 instance with SSM permissions"
# Create trust policy file for EC2
cat > ec2-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role for EC2
```

```
echo "Creating IAM role $EC2_ROLE_NAME for EC2 instance..."
EC2_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$EC2_ROLE_NAME" \
  --assume-role-policy-document file://ec2-trust-policy.json)
check_error "$EC2_ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$EC2_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $EC2_ROLE_NAME")

# Attach SSM policy to the EC2 role
echo "Attaching AmazonSSMManagedInstanceCore policy to role $EC2_ROLE_NAME..."
EC2_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$EC2_ROLE_NAME" \
  --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore)
check_error "$EC2_POLICY_OUTPUT" "aws iam attach-role-policy"
CREATED_RESOURCES+=("IAM Policy: AmazonSSMManagedInstanceCore attached to
 $EC2_ROLE_NAME")

# Create instance profile
echo "Creating instance profile $INSTANCE_PROFILE_NAME..."
PROFILE_OUTPUT=$(aws iam create-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME")
check_error "$PROFILE_OUTPUT" "aws iam create-instance-profile"
CREATED_RESOURCES+=("IAM Instance Profile: $INSTANCE_PROFILE_NAME")

# Add role to instance profile
echo "Adding role $EC2_ROLE_NAME to instance profile $INSTANCE_PROFILE_NAME..."
ADD_ROLE_OUTPUT=$(aws iam add-role-to-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME" \
  --role-name "$EC2_ROLE_NAME")
check_error "$ADD_ROLE_OUTPUT" "aws iam add-role-to-instance-profile"

# Wait for role to propagate
echo "Waiting for IAM role to propagate..."
sleep 10

echo "Step 3: Launching EC2 instance"
# Get the latest Amazon Linux 2 AMI ID
echo "Finding latest Amazon Linux 2 AMI..."
AMI_ID=$(aws ec2 describe-images \
  --owners amazon \
  --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
  --query "sort_by(Images, &CreationDate)[-1].ImageId" \
```

```
  --output text)
check_error "$AMI_ID" "aws ec2 describe-images"
echo "Using AMI: $AMI_ID"

# Launch EC2 instance
echo "Launching EC2 instance with AMI $AMI_ID..."
INSTANCE_OUTPUT=$(aws ec2 run-instances \
  --image-id "$AMI_ID" \
  --instance-type t2.micro \
  --iam-instance-profile Name="$INSTANCE_PROFILE_NAME" \
  --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=FIS-Test-
Instance},{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-fault-injection-
service-gs}]')
check_error "$INSTANCE_OUTPUT" "aws ec2 run-instances"

# Get instance ID
INSTANCE_ID=$(echo "$INSTANCE_OUTPUT" | grep -i "InstanceId" | head -1 | awk -
F'"' '{print $4}')
if [ -z "$INSTANCE_ID" ]; then
    echo "Failed to get instance ID"
    cleanup_on_error
    exit 1
fi
echo "Launched instance: $INSTANCE_ID"
CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")

# Enable detailed monitoring
echo "Enabling detailed monitoring for instance $INSTANCE_ID..."
MONITOR_OUTPUT=$(aws ec2 monitor-instances --instance-ids "$INSTANCE_ID")
check_error "$MONITOR_OUTPUT" "aws ec2 monitor-instances"

# Wait for instance to be running and status checks to pass
echo "Waiting for instance to be ready..."
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"
aws ec2 wait instance-status-ok --instance-ids "$INSTANCE_ID"
echo "Instance is ready"

echo "Step 4: Creating CloudWatch alarm for CPU utilization"
# Create CloudWatch alarm
echo "Creating CloudWatch alarm $ALARM_NAME..."
ALARM_OUTPUT=$(aws cloudwatch put-metric-alarm \
  --alarm-name "$ALARM_NAME" \
  --alarm-description "Alarm when CPU exceeds 50%" \
  --metric-name CPUUtilization \
```

```
  --namespace AWS/EC2 \
  --statistic Maximum \
  --period 60 \
  --threshold 50 \
  --comparison-operator GreaterThanOrEqualToThreshold \
  --dimensions "Name=InstanceId,Value=$INSTANCE_ID" \
  --evaluation-periods 1)
check_error "$ALARM_OUTPUT" "aws cloudwatch put-metric-alarm"
CREATED_RESOURCES+=("CloudWatch Alarm: $ALARM_NAME")

# Get the alarm ARN
echo "Getting CloudWatch alarm ARN..."
ALARM_ARN_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
check_error "$ALARM_ARN_OUTPUT" "aws cloudwatch describe-alarms"
ALARM_ARN=$(echo "$ALARM_ARN_OUTPUT" | grep -i "AlarmArn" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$ALARM_ARN" ]; then
    echo "Failed to get alarm ARN"
    cleanup_on_error
    exit 1
fi
echo "Alarm ARN: $ALARM_ARN"

# Wait for the alarm to initialize and reach OK state
echo "Waiting for CloudWatch alarm to initialize (60 seconds)..."
sleep 60

# Check alarm state
echo "Checking alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 | awk -
F'"' '{print $4}')
echo "Current alarm state: $ALARM_STATE"

# If alarm is not in OK state, wait longer or generate some baseline metrics
if [ "$ALARM_STATE" != "OK" ]; then
    echo "Alarm not in OK state. Waiting for alarm to stabilize (additional 60
 seconds)..."
    sleep 60

    # Check alarm state again
    ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
```

```
      --alarm-names "$ALARM_NAME")
    ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 |
 awk -F'"' '{print $4}')
    echo "Updated alarm state: $ALARM_STATE"

    if [ "$ALARM_STATE" != "OK" ]; then
        echo "Warning: Alarm still not in OK state. Experiment may fail to
 start."
    fi
fi

echo "Step 5: Creating AWS FIS experiment template"
# Get the IAM role ARN
echo "Getting IAM role ARN for $FIS_ROLE_NAME..."
ROLE_ARN_OUTPUT=$(aws iam get-role \
  --role-name "$FIS_ROLE_NAME")
check_error "$ROLE_ARN_OUTPUT" "aws iam get-role"
ROLE_ARN=$(echo "$ROLE_ARN_OUTPUT" | grep -i "Arn" | head -1 | awk -F'"' '{print
 $4}')
if [ -z "$ROLE_ARN" ]; then
    echo "Failed to get role ARN"
    cleanup_on_error
    exit 1
fi
echo "Role ARN: $ROLE_ARN"

# Get account ID and region
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"  # Default to us-east-1 if region not set
fi
INSTANCE_ARN="arn:aws:ec2:${REGION}:${ACCOUNT_ID}:instance/${INSTANCE_ID}"
echo "Instance ARN: $INSTANCE_ARN"

# Create experiment template - Fixed JSON escaping issue
cat > experiment-template.json << EOF
{
  "description": "Test CPU stress predefined SSM document",
  "targets": {
    "testInstance": {
      "resourceType": "aws:ec2:instance",
      "resourceArns": ["$INSTANCE_ARN"],
      "selectionMode": "ALL"
```

```
    }
  },
  "actions": {
    "runCpuStress": {
      "actionId": "aws:ssm:send-command",
      "parameters": {
        "documentArn": "arn:aws:ssm:$REGION::document/AWSFIS-Run-CPU-Stress",
        "documentParameters": "{\"DurationSeconds\":\"120\"}",
        "duration": "PT5M"
      },
      "targets": {
        "Instances": "testInstance"
      }
    }
  },
  "stopConditions": [
    {
      "source": "aws:cloudwatch:alarm",
      "value": "$ALARM_ARN"
    }
  ],
  "roleArn": "$ROLE_ARN",
  "tags": {
    "Name": "FIS-CPU-Stress-Experiment",
    "project": "doc-smith",
    "tutorial": "aws-fault-injection-service-gs"
  }
}
EOF

# Create experiment template
echo "Creating AWS FIS experiment template..."
TEMPLATE_OUTPUT=$(aws fis create-experiment-template --cli-input-json file://
experiment-template.json)
check_error "$TEMPLATE_OUTPUT" "aws fis create-experiment-template"
TEMPLATE_ID=$(echo "$TEMPLATE_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$TEMPLATE_ID" ]; then
    echo "Failed to get template ID"
    cleanup_on_error
    exit 1
fi
echo "Experiment template created with ID: $TEMPLATE_ID"
CREATED_RESOURCES+=("FIS Experiment Template: $TEMPLATE_ID")
```

```
echo "Step 6: Starting the experiment"
# Start the experiment
echo "Starting AWS FIS experiment using template $TEMPLATE_ID..."
EXPERIMENT_OUTPUT=$(aws fis start-experiment \
  --experiment-template-id "$TEMPLATE_ID" \
  --tags '{"Name": "FIS-CPU-Stress-Run"}')
check_error "$EXPERIMENT_OUTPUT" "aws fis start-experiment"
EXPERIMENT_ID=$(echo "$EXPERIMENT_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$EXPERIMENT_ID" ]; then
    echo "Failed to get experiment ID"
    cleanup_on_error
    exit 1
fi
echo "Experiment started with ID: $EXPERIMENT_ID"
CREATED_RESOURCES+=("FIS Experiment: $EXPERIMENT_ID")

echo "Step 7: Tracking experiment progress"
# Track experiment progress
echo "Tracking experiment progress..."
MAX_CHECKS=30
CHECK_COUNT=0
EXPERIMENT_STATE=""

while [ $CHECK_COUNT -lt $MAX_CHECKS ]; do
    EXPERIMENT_INFO=$(aws fis get-experiment --id "$EXPERIMENT_ID")
    # Don't check for errors here, as we expect some experiments to fail

    EXPERIMENT_STATE=$(echo "$EXPERIMENT_INFO" | grep -i "status" | head -1 | awk
 -F'"' '{print $4}')
    echo "Experiment state: $EXPERIMENT_STATE"

    if [ "$EXPERIMENT_STATE" == "completed" ] || [ "$EXPERIMENT_STATE" ==
 "stopped" ] || [ "$EXPERIMENT_STATE" == "failed" ]; then
        # Show the reason for the state
        REASON=$(echo "$EXPERIMENT_INFO" | grep -i "reason" | head -1 | awk -F'"'
 '{print $4}')
        if [ -n "$REASON" ]; then
            echo "Reason: $REASON"
        fi
        break
    fi
```

```
    echo "Waiting 10 seconds before checking again..."
    sleep 10
    CHECK_COUNT=$((CHECK_COUNT + 1))
done

if [ $CHECK_COUNT -eq $MAX_CHECKS ]; then
    echo "Experiment is taking longer than expected. You can check its status
 later using:"
    echo "aws fis get-experiment --id $EXPERIMENT_ID"
fi

echo "Step 8: Verifying experiment results"
# Check CloudWatch alarm state
echo "Checking CloudWatch alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms --alarm-names "$ALARM_NAME")
check_error "$ALARM_STATE_OUTPUT" "aws cloudwatch describe-alarms"
echo "$ALARM_STATE_OUTPUT"

# Get CPU utilization metrics
echo "Getting CPU utilization metrics..."
END_TIME=$(date -u +"%Y-%m-%dT%H:%M:%SZ")

# FIXED: Cross-platform compatible way to calculate time 10 minutes ago
# This approach uses epoch seconds and basic arithmetic which works on all Linux
 distributions
CURRENT_EPOCH=$(date +%s)
TEN_MINUTES_AGO_EPOCH=$((CURRENT_EPOCH - 600))
START_TIME=$(date -u -d "@$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ" 2>/dev/
null || date -u -r "$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ")

# Create metric query file
cat > metric-query.json << EOF
[
  {
    "Id": "cpu",
    "MetricStat": {
      "Metric": {
        "Namespace": "AWS/EC2",
        "MetricName": "CPUUtilization",
        "Dimensions": [
          {
            "Name": "InstanceId",
            "Value": "$INSTANCE_ID"
          }
```

```
        ]
      },
      "Period": 60,
      "Stat": "Maximum"
    }
  }
]
EOF

METRICS_OUTPUT=$(aws cloudwatch get-metric-data \
  --start-time "$START_TIME" \
  --end-time "$END_TIME" \
  --metric-data-queries file://metric-query.json)
check_error "$METRICS_OUTPUT" "aws cloudwatch get-metric-data"
echo "CPU Utilization Metrics:"
echo "$METRICS_OUTPUT"

# Display summary of created resources
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource"
done
echo "==========================================="

# Prompt for cleanup
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Starting cleanup process..."

    # Stop experiment if still running
    if [ "$EXPERIMENT_STATE" != "completed" ] && [ "$EXPERIMENT_STATE" !=
 "stopped" ] && [ "$EXPERIMENT_STATE" != "failed" ]; then
        echo "Stopping experiment $EXPERIMENT_ID..."
        STOP_OUTPUT=$(aws fis stop-experiment --id "$EXPERIMENT_ID")
        check_error "$STOP_OUTPUT" "aws fis stop-experiment"
```

```
        echo "Waiting for experiment to stop..."
        sleep 10
    fi

    # Delete experiment template
    echo "Deleting experiment template $TEMPLATE_ID..."
    DELETE_TEMPLATE_OUTPUT=$(aws fis delete-experiment-template --id
"$TEMPLATE_ID")
    check_error "$DELETE_TEMPLATE_OUTPUT" "aws fis delete-experiment-template"

    # Delete CloudWatch alarm
    echo "Deleting CloudWatch alarm $ALARM_NAME..."
    DELETE_ALARM_OUTPUT=$(aws cloudwatch delete-alarms --alarm-names
"$ALARM_NAME")
    check_error "$DELETE_ALARM_OUTPUT" "aws cloudwatch delete-alarms"

    # Terminate EC2 instance
    echo "Terminating EC2 instance $INSTANCE_ID..."
    TERMINATE_OUTPUT=$(aws ec2 terminate-instances --instance-ids "$INSTANCE_ID")
    check_error "$TERMINATE_OUTPUT" "aws ec2 terminate-instances"
    echo "Waiting for instance to terminate..."
    aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"

    # Clean up IAM resources
    echo "Removing role from instance profile..."
    REMOVE_ROLE_OUTPUT=$(aws iam remove-role-from-instance-profile \
      --instance-profile-name "$INSTANCE_PROFILE_NAME" \
      --role-name "$EC2_ROLE_NAME")
    check_error "$REMOVE_ROLE_OUTPUT" "aws iam remove-role-from-instance-profile"

    echo "Deleting instance profile..."
    DELETE_PROFILE_OUTPUT=$(aws iam delete-instance-profile \
      --instance-profile-name "$INSTANCE_PROFILE_NAME")
    check_error "$DELETE_PROFILE_OUTPUT" "aws iam delete-instance-profile"

    echo "Deleting FIS role policy..."
    DELETE_POLICY_OUTPUT=$(aws iam delete-role-policy \
      --role-name "$FIS_ROLE_NAME" \
      --policy-name "$FIS_POLICY_NAME")
    check_error "$DELETE_POLICY_OUTPUT" "aws iam delete-role-policy"

    echo "Detaching policy from EC2 role..."
    DETACH_POLICY_OUTPUT=$(aws iam detach-role-policy \
      --role-name "$EC2_ROLE_NAME" \
```

```
      --policy-arn "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore")
    check_error "$DETACH_POLICY_OUTPUT" "aws iam detach-role-policy"

    echo "Deleting FIS role..."
    DELETE_FIS_ROLE_OUTPUT=$(aws iam delete-role \
      --role-name "$FIS_ROLE_NAME")
    check_error "$DELETE_FIS_ROLE_OUTPUT" "aws iam delete-role"

    echo "Deleting EC2 role..."
    DELETE_EC2_ROLE_OUTPUT=$(aws iam delete-role \
      --role-name "$EC2_ROLE_NAME")
    check_error "$DELETE_EC2_ROLE_OUTPUT" "aws iam delete-role"

    # Clean up temporary files
    echo "Cleaning up temporary files..."
    rm -f fis-trust-policy.json ec2-trust-policy.json fis-ssm-policy.json
 experiment-template.json metric-query.json

    echo "Cleanup completed successfully."
else
    echo "Cleanup skipped. Resources will remain in your AWS account."
    echo "You can manually clean up the resources listed above."
fi

echo ""
echo "Script execution completed."
echo "Log file: $LOG_FILE"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - CreateExperimentTemplate
  - CreateInstanceProfile
  - CreateRole
  - DeleteAlarms
  - DeleteExperimentTemplate
  - DeleteInstanceProfile

- DeleteRole
- DeleteRolePolicy
- DescribeAlarms
- DescribeImages
- DetachRolePolicy
- GetCallerIdentity
- GetExperiment
- GetMetricData
- GetRole
- MonitorInstances
- PutMetricAlarm
- PutRolePolicy
- RemoveRoleFromInstanceProfile
- RunInstances
- StartExperiment
- StopExperiment
- TerminateInstances
- Attente

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Configurer le contrôle Attribute-Based d'accès pour DynamoDB à l'aide de AWS Command Line Interface v2

L'exemple de code suivant montre comment implémenter le contrôle Attribute-Based d'accès (ABAC) pour DynamoDB.

- Créez une politique IAM pour ABAC.
- Créez des tables avec des balises pour différents départements.
- Répertoriez et filtrez les tables en fonction des balises.

Bash

AWS CLI avec le script Bash

Créez une politique IAM pour ABAC.

```
# Step 1: Create a policy document for ABAC
cat > abac-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetItem",
        "dynamodb:BatchGetItem",
        "dynamodb:Query",
        "dynamodb:Scan"
      ],
      "Resource": "arn:aws:dynamodb:*:*:table/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Department": "${aws:PrincipalTag/Department}"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:PutItem",
        "dynamodb:UpdateItem",
        "dynamodb:DeleteItem",
        "dynamodb:BatchWriteItem"
      ],
      "Resource": "arn:aws:dynamodb:*:*:table/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Department": "${aws:PrincipalTag/Department}",
          "aws:ResourceTag/Environment": "Development"
        }
      }
    }
  ]
}
```

```
EOF

# Step 2: Create the IAM policy
aws iam create-policy \
    --policy-name DynamoDBDepartmentBasedAccess \
    --policy-document file://abac-policy.json
```

Créez des tables avec des balises pour différents départements.

```
# Create a DynamoDB table with tags for ABAC
aws dynamodb create-table \
    --table-name FinanceData \
    --attribute-definitions \
        AttributeName=RecordID,AttributeType=S \
    --key-schema \
        AttributeName=RecordID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --tags \
        Key=Department,Value=Finance \
        Key=Environment,Value=Development

# Create another table with different tags
aws dynamodb create-table \
    --table-name MarketingData \
    --attribute-definitions \
        AttributeName=RecordID,AttributeType=S \
    --key-schema \
        AttributeName=RecordID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --tags \
        Key=Department,Value=Marketing \
        Key=Environment,Value=Production
```

Répertoriez et filtrez les tables en fonction des balises.

```
# List all DynamoDB tables
echo "Listing all tables:"
aws dynamodb list-tables

# Get ARNs for all tables
echo -e "\nGetting ARNs for all tables:"
```

```
TABLE_ARNS=$(aws dynamodb list-tables --query "TableNames[*]" --output text |
 xargs -I {} aws dynamodb describe-table --table-name {} --query "Table.TableArn"
 --output text)

# For each table ARN, list its tags
echo -e "\nListing tags for each table:"
for ARN in $TABLE_ARNS; do
    TABLE_NAME=$(echo $ARN | awk -F/ '{print $2}')
    echo -e "\nTags for table: $TABLE_NAME"
    aws dynamodb list-tags-of-resource --resource-arn $ARN
done

# Example: Find tables with a specific tag
echo -e "\nFinding tables with Environment=Production tag:"
for ARN in $TABLE_ARNS; do
    TABLE_NAME=$(echo $ARN | awk -F/ '{print $2}')
    TAGS=$(aws dynamodb list-tags-of-resource --resource-arn $ARN --query "Tags[?
Key=='Environment' && Value=='Production']" --output text)
    if [ ! -z "$TAGS" ]; then
        echo "Table with Production tag: $TABLE_NAME"
    fi
done
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - CreatePolicy
  - CreateTable
  - ListTables

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Configuration de la gestion des systèmes

L'exemple de code suivant illustre comment :

- Création d'autorisations IAM pour Systems Manager
- Création d'un rôle IAM pour Systems Manager

- Configuration de Systems Manager
- Vérifiez la configuration
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# AWS Systems Manager Setup Script
# This script sets up AWS Systems Manager for a single account and region
#
# Version 17 fixes:
# 1. Added cloudformation.amazonaws.com to the IAM role trust policy
# 2. Systems Manager Quick Setup uses CloudFormation for deployments, so the role
 must trust CloudFormation service

# Initialize log file
LOG_FILE="ssm_setup_$(date +%Y%m%d_%H%M%S).log"
UNIQUE_ID=$(openssl rand -hex 4)
echo "Starting AWS Systems Manager setup at $(date)" > "$LOG_FILE"

# Function to log commands and their outputs with immediate terminal display
log_cmd() {
    echo "$(date): Running command: $1" | tee -a "$LOG_FILE"
    local output
    output=$(eval "$1" 2>&1)
    local status=$?
    echo "$output" | tee -a "$LOG_FILE"
    return $status
}

# Function to check for errors in command output
check_error() {
```

```
    local cmd_output="$1"
    local cmd_status="$2"
    local error_msg="$3"

    if [[ $cmd_status -ne 0 || "$cmd_output" =~ [Ee][Rr][Rr][Oo][Rr] ]]; then
        echo "ERROR: $error_msg" | tee -a "$LOG_FILE"
        echo "Command output: $cmd_output" | tee -a "$LOG_FILE"
        cleanup_on_error
        exit 1
    fi
}

# Array to track created resources for cleanup
declare -a CREATED_RESOURCES

# Function to add a resource to the tracking array
track_resource() {
    local resource_type="$1"
    local resource_id="$2"
    CREATED_RESOURCES+=("$resource_type:$resource_id")
    echo "Tracked resource: $resource_type:$resource_id" | tee -a "$LOG_FILE"
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "" | tee -a "$LOG_FILE"
    echo "==========================================" | tee -a "$LOG_FILE"
    echo "ERROR OCCURRED - CLEANING UP RESOURCES" | tee -a "$LOG_FILE"
    echo "==========================================" | tee -a "$LOG_FILE"
    echo "The following resources were created:" | tee -a "$LOG_FILE"

    # Display resources in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        echo "${CREATED_RESOURCES[$i]}" | tee -a "$LOG_FILE"
    done

    echo "" | tee -a "$LOG_FILE"
    echo "Attempting to clean up resources..." | tee -a "$LOG_FILE"

    # Clean up resources in reverse order
    cleanup_resources
}

# Function to clean up all created resources
```

```
cleanup_resources() {
    # Process resources in reverse order (last created, first deleted)
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        IFS=':' read -r resource_type resource_id <<< "${CREATED_RESOURCES[$i]}"

        echo "Deleting $resource_type: $resource_id" | tee -a "$LOG_FILE"

        case "$resource_type" in
            "IAM_POLICY")
                # Delete the policy (detachment should have been handled when the
 role was deleted)
                log_cmd "aws iam delete-policy --policy-arn $resource_id" || true
                ;;
            "IAM_ROLE")
                # Detach all policies from the role first
                if [[ -n "$POLICY_ARN" ]]; then
                    log_cmd "aws iam detach-role-policy --role-name $resource_id
 --policy-arn $POLICY_ARN" || true
                fi

                # Delete the role
                log_cmd "aws iam delete-role --role-name $resource_id" || true
                ;;
            "SSM_CONFIG_MANAGER")
                log_cmd "aws ssm-quicksetup delete-configuration-manager --
manager-arn $resource_id" || true
                ;;
            *)
                echo "Unknown resource type: $resource_type, cannot delete
 automatically" | tee -a "$LOG_FILE"
                ;;
        esac
    done

    echo "Cleanup completed" | tee -a "$LOG_FILE"

    # Clean up temporary files
    rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/
null || true
}

# Main script execution
echo "AWS Systems Manager Setup Script"
echo "================================"
```

```
echo "This script will set up AWS Systems Manager for a single account and
 region."
echo "It will create IAM policies and roles, then enable Systems Manager
 features."
echo ""

# Get the current AWS region
CURRENT_REGION=$(aws configure get region)
if [[ -z "$CURRENT_REGION" ]]; then
    echo "No AWS region configured. Please specify a region:"
    CURRENT_REGION="${AWS_DEFAULT_REGION:-us-west-2}"
    if [[ -z "$CURRENT_REGION" ]]; then
        echo "ERROR: A region must be specified" | tee -a "$LOG_FILE"
        exit 1
    fi
fi

echo "Using AWS region: $CURRENT_REGION" | tee -a "$LOG_FILE"

# Step 1: Create IAM policy for Systems Manager onboarding
echo "Step 1: Creating IAM policy for Systems Manager onboarding..."

# Create policy document
cat > ssm-onboarding-policy.json << 'EOF'
{
   "Version":"2012-10-17",
   "Statement": [
     {
       "Sid": "QuickSetupActions",
       "Effect": "Allow",
       "Action": [
         "ssm-quicksetup:*"
       ],
       "Resource": "*"
     },
     {
       "Sid": "SsmReadOnly",
       "Effect": "Allow",
       "Action": [
         "ssm:DescribeAutomationExecutions",
         "ssm:GetAutomationExecution",
         "ssm:ListAssociations",
         "ssm:DescribeAssociation",
         "ssm:ListDocuments",
```

```
            "ssm:ListResourceDataSync",
            "ssm:DescribePatchBaselines",
            "ssm:GetPatchBaseline",
            "ssm:DescribeMaintenanceWindows",
            "ssm:DescribeMaintenanceWindowTasks"
        ],
        "Resource": "*"
    },
    {
        "Sid": "SsmDocument",
        "Effect": "Allow",
        "Action": [
            "ssm:GetDocument",
            "ssm:DescribeDocument"
        ],
        "Resource": [
            "arn:aws:ssm:*:*:document/AWSQuickSetupType-*",
            "arn:aws:ssm:*:*:document/AWS-EnableExplorer"
        ]
    },
    {
        "Sid": "SsmEnableExplorer",
        "Effect": "Allow",
        "Action": "ssm:StartAutomationExecution",
        "Resource": "arn:aws:ssm:*:*:automation-definition/AWS-EnableExplorer:*"
    },
    {
        "Sid": "SsmExplorerRds",
        "Effect": "Allow",
        "Action": [
            "ssm:GetOpsSummary",
            "ssm:CreateResourceDataSync",
            "ssm:UpdateResourceDataSync"
        ],
        "Resource": "arn:aws:ssm:*:*:resource-data-sync/AWS-QuickSetup-*"
    },
    {
        "Sid": "OrgsReadOnly",
        "Effect": "Allow",
        "Action": [
            "organizations:DescribeAccount",
            "organizations:DescribeOrganization",
            "organizations:ListDelegatedAdministrators",
            "organizations:ListRoots",
```

```
        "organizations:ListParents",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAWSServiceAccessForOrganization"
      ],
      "Resource": "*"
    },
    {
      "Sid": "OrgsAdministration",
      "Effect": "Allow",
      "Action": [
        "organizations:EnableAWSServiceAccess",
        "organizations:RegisterDelegatedAdministrator",
        "organizations:DeregisterDelegatedAdministrator"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "organizations:ServicePrincipal": [
            "ssm.amazonaws.com",
            "ssm-quicksetup.amazonaws.com",
            "member.org.stacksets.cloudformation.amazonaws.com",
            "resource-explorer-2.amazonaws.com"
          ]
        }
      }
    },
    {
      "Sid": "CfnReadOnly",
      "Effect": "Allow",
      "Action": [
        "cloudformation:ListStacks",
        "cloudformation:DescribeStacks",
        "cloudformation:ListStackSets",
        "cloudformation:DescribeOrganizationsAccess"
      ],
      "Resource": "*"
    },
    {
      "Sid": "OrgCfnAccess",
      "Effect": "Allow",
      "Action": [
        "cloudformation:ActivateOrganizationsAccess"
      ],
```

```
      "Resource": "*"
    },
    {
      "Sid": "CfnStackActions",
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStack",
        "cloudformation:DeleteStack",
        "cloudformation:DescribeStackResources",
        "cloudformation:DescribeStackEvents",
        "cloudformation:GetTemplate",
        "cloudformation:RollbackStack",
        "cloudformation:TagResource",
        "cloudformation:UntagResource",
        "cloudformation:UpdateStack"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stack/StackSet-AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:stack/AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:type/resource/*"
      ]
    },
    {
      "Sid": "CfnStackSetActions",
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStackInstances",
        "cloudformation:CreateStackSet",
        "cloudformation:DeleteStackInstances",
        "cloudformation:DeleteStackSet",
        "cloudformation:DescribeStackInstance",
        "cloudformation:DetectStackSetDrift",
        "cloudformation:ListStackInstanceResourceDrifts",
        "cloudformation:DescribeStackSet",
        "cloudformation:DescribeStackSetOperation",
        "cloudformation:ListStackInstances",
        "cloudformation:ListStackSetOperations",
        "cloudformation:ListStackSetOperationResults",
        "cloudformation:TagResource",
        "cloudformation:UntagResource",
        "cloudformation:UpdateStackSet"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-*",
```

```
        "arn:aws:cloudformation:*:*:type/resource/*",
        "arn:aws:cloudformation:*:*:stackset-target/AWS-QuickSetup-*:*"
      ]
    },
    {
      "Sid": "ValidationReadonlyActions",
      "Effect": "Allow",
      "Action": [
        "iam:ListRoles",
        "iam:GetRole"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IamRolesMgmt",
      "Effect": "Allow",
      "Action": [
        "iam:CreateRole",
        "iam:DeleteRole",
        "iam:GetRole",
        "iam:AttachRolePolicy",
        "iam:DetachRolePolicy",
        "iam:GetRolePolicy",
        "iam:ListRolePolicies"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ]
    },
    {
      "Sid": "IamPassRole",
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ],
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": [
            "ssm.amazonaws.com",
```

```
            "ssm-quicksetup.amazonaws.com",
            "cloudformation.amazonaws.com"
          ]
        }
      }
    },
    {
      "Sid": "IamRolesPoliciesMgmt",
      "Effect": "Allow",
      "Action": [
        "iam:AttachRolePolicy",
        "iam:DetachRolePolicy"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ],
      "Condition": {
        "ArnEquals": {
          "iam:PolicyARN": [
            "arn:aws:iam::aws:policy/
AWSSystemsManagerEnableExplorerExecutionPolicy",
            "arn:aws:iam::aws:policy/AWSQuickSetupSSMDeploymentRolePolicy"
          ]
        }
      }
    },
    {
      "Sid": "CfnStackSetsSLR",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/aws-service-role/
stacksets.cloudformation.amazonaws.com/
AWSServiceRoleForCloudFormationStackSetsOrgAdmin",
        "arn:aws:iam::*:role/aws-service-role/ssm.amazonaws.com/
AWSServiceRoleForAmazonSSM",
        "arn:aws:iam::*:role/aws-service-role/
accountdiscovery.ssm.amazonaws.com/AWSServiceRoleForAmazonSSM_AccountDiscovery",
        "arn:aws:iam::*:role/aws-service-role/ssm-quicksetup.amazonaws.com/
AWSServiceRoleForSSMQuickSetup",
```

```
        "arn:aws:iam::*:role/aws-service-role/resource-explorer-2.amazonaws.com/
AWSServiceRoleForResourceExplorer"
      ]
    }
  ]
}
EOF

# Create the IAM policy
POLICY_OUTPUT=$(log_cmd "aws iam create-policy --policy-name SSMOnboardingPolicy-
$UNIQUE_ID --policy-document file://ssm-onboarding-policy.json --output json")
POLICY_STATUS=$?
check_error "$POLICY_OUTPUT" $POLICY_STATUS "Failed to create IAM policy"

# Extract the policy ARN
POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:policy/
SSMOnboardingPolicy-[a-f0-9]*')
if [[ -z "$POLICY_ARN" ]]; then
    echo "ERROR: Failed to extract policy ARN" | tee -a "$LOG_FILE"
    exit 1
fi

# Track the created policy
track_resource "IAM_POLICY" "$POLICY_ARN"
aws iam tag-policy --policy-arn "$POLICY_ARN" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created policy: $POLICY_ARN" | tee -a "$LOG_FILE"

# Step 2: Create and configure IAM role for Systems Manager
echo ""
echo "Step 2: Creating IAM role for Systems Manager..."

# Get current user name
USER_OUTPUT=$(log_cmd "aws sts get-caller-identity --output json")
USER_STATUS=$?
check_error "$USER_OUTPUT" $USER_STATUS "Failed to get caller identity"

# Extract account ID
ACCOUNT_ID=$(echo "$USER_OUTPUT" | grep -o '"Account": "[0-9]*"' | cut -d'"' -f4)
if [[ -z "$ACCOUNT_ID" ]]; then
    echo "ERROR: Failed to extract account ID" | tee -a "$LOG_FILE"
    exit 1
fi
```

```
# Generate a unique role name
ROLE_NAME="SSMTutorialRole-$(openssl rand -hex 4)"

# Create trust policy for the role - FIXED: Added cloudformation.amazonaws.com
cat > trust-policy.json << 'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com",
                    "ssm-quicksetup.amazonaws.com",
                    "cloudformation.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::ACCOUNT_ID:root"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

# Replace ACCOUNT_ID placeholder in trust policy
sed -i "s/ACCOUNT_ID/$ACCOUNT_ID/g" trust-policy.json

# Create the IAM role
ROLE_OUTPUT=$(log_cmd "aws iam create-role --role-name $ROLE_NAME --assume-role-
policy-document file://trust-policy.json --description 'Role for Systems Manager
 tutorial' --output json")
ROLE_STATUS=$?
check_error "$ROLE_OUTPUT" $ROLE_STATUS "Failed to create IAM role"

# Extract the role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:role/[^"]*')
if [[ -z "$ROLE_ARN" ]]; then
```

```
    echo "ERROR: Failed to extract role ARN" | tee -a "$LOG_FILE"
    cleanup_on_error
    exit 1
fi

# Track the created role
track_resource "IAM_ROLE" "$ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created IAM role: $ROLE_NAME" | tee -a "$LOG_FILE"
echo "Role ARN: $ROLE_ARN" | tee -a "$LOG_FILE"

# Set identity variables for cleanup
IDENTITY_TYPE="role"
IDENTITY_NAME="$ROLE_NAME"

# Attach the policy to the role
ATTACH_OUTPUT=$(log_cmd "aws iam attach-role-policy --role-name $ROLE_NAME --
policy-arn $POLICY_ARN")
ATTACH_STATUS=$?
check_error "$ATTACH_OUTPUT" $ATTACH_STATUS "Failed to attach policy to role
 $ROLE_NAME"

echo "Policy attached to role: $ROLE_NAME" | tee -a "$LOG_FILE"

# Step 3: Create Systems Manager configuration using Host Management
echo ""
echo "Step 3: Creating Systems Manager configuration..."

# Generate a random identifier for the configuration name
CONFIG_NAME="SSMSetup-$(openssl rand -hex 4)"

# Create configuration file for Systems Manager setup using Host Management
# Added both required parameters for single account deployment based on
 CloudFormation documentation
cat > ssm-config.json << EOF
[
  {
    "Type": "AWSQuickSetupType-SSMHostMgmt",
    "LocalDeploymentAdministrationRoleArn": "$ROLE_ARN",
    "LocalDeploymentExecutionRoleName": "$ROLE_NAME",
    "Parameters": {
      "TargetAccounts": "$ACCOUNT_ID",
```

```
      "TargetRegions": "$CURRENT_REGION"
    }
  }
]
EOF

echo "Configuration file created:" | tee -a "$LOG_FILE"
cat ssm-config.json | tee -a "$LOG_FILE"

# Create the configuration manager
CONFIG_OUTPUT=$(log_cmd "aws ssm-quicksetup create-configuration-manager --name
 \"$CONFIG_NAME\" --configuration-definitions file://ssm-config.json --region
 $CURRENT_REGION")
CONFIG_STATUS=$?
check_error "$CONFIG_OUTPUT" $CONFIG_STATUS "Failed to create Systems Manager
 configuration"

# Extract the manager ARN
MANAGER_ARN=$(echo "$CONFIG_OUTPUT" | grep -o 'arn:aws:ssm-quicksetup:[^"]*')
if [[ -z "$MANAGER_ARN" ]]; then
    echo "ERROR: Failed to extract manager ARN" | tee -a "$LOG_FILE"
    exit 1
fi

# Track the created configuration manager
track_resource "SSM_CONFIG_MANAGER" "$MANAGER_ARN"

echo "Created Systems Manager configuration: $MANAGER_ARN" | tee -a "$LOG_FILE"

# Step 4: Verify the setup
echo ""
echo "Step 4: Verifying the setup..."

# Wait for the configuration to be fully deployed
echo "Waiting for the configuration to be deployed (this may take a few
 minutes)..."
sleep 30

# Check the configuration manager status
VERIFY_OUTPUT=$(log_cmd "aws ssm-quicksetup get-configuration-manager --manager-
arn $MANAGER_ARN --region $CURRENT_REGION")
VERIFY_STATUS=$?
check_error "$VERIFY_OUTPUT" $VERIFY_STATUS "Failed to verify configuration
 manager"
```

```
echo "Systems Manager setup completed successfully!" | tee -a "$LOG_FILE"

# List the created resources
echo ""
echo "==========================================="
echo "CREATED RESOURCES"
echo "==========================================="
for resource in "${CREATED_RESOURCES[@]+"${CREATED_RESOURCES[@]}"}"; do
    echo "$resource"
done

# Prompt for cleanup
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE="y"

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..." | tee -a "$LOG_FILE"
    cleanup_resources
    echo "Cleanup completed." | tee -a "$LOG_FILE"
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later." | tee -a "$LOG_FILE"
fi

echo ""
echo "Script execution completed. See $LOG_FILE for details."

# Clean up temporary files
rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/null ||
 true
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreateConfigurationManager
  - CreatePolicy

- CreateRole
- DeleteConfigurationManager
- DeletePolicy
- DeleteRole
- DetachRolePolicy
- GetCallerIdentity
- GetConfigurationManager

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de variables de propriété dans les tableaux de bord de surveillance pour surveiller plusieurs fonctions sans serveur

L'exemple de code suivant illustre comment :

- Création de fonctions Lambda pour la surveillance
- Création d'un CloudWatch tableau de bord
- Ajouter une variable de propriété au tableau de bord
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# CloudWatch Dashboard with Lambda Function Variable Script
```

```
# This script creates a CloudWatch dashboard with a property variable for Lambda
 function names

set -euo pipefail

# Security: Set restrictive umask
umask 0077

# Set up logging with secure permissions
LOG_FILE="cloudwatch-dashboard-script-v4.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"
echo "Starting script execution at $(date)" >> "$LOG_FILE"

# Function to log commands and their output (with sensitive data sanitization)
log_cmd() {
    local cmd="$1"
    local sanitized_cmd="${cmd//--password*/--password [REDACTED]}"
    sanitized_cmd="${sanitized_cmd//--secret*/--secret [REDACTED]}"
    echo "$(date): Running command: $sanitized_cmd" >> "$LOG_FILE"
    eval "$cmd" 2>&1 | tee -a "$LOG_FILE"
    return ${PIPESTATUS[0]}
}

# Function to check for errors in command output
check_error() {
    local cmd_output="$1"
    local cmd_status="$2"
    local error_msg="$3"

    if [ $cmd_status -ne 0 ] || echo "$cmd_output" | grep -qi "error"; then
        echo "ERROR: $error_msg" | tee -a "$LOG_FILE"
        # Sanitize output before logging
        local sanitized_output="${cmd_output//arn:aws:iam::[0-9]*/
arn:aws:iam::ACCOUNT_ID}"
        echo "Command output: $sanitized_output" | tee -a "$LOG_FILE"
        cleanup_resources
        exit 1
    fi
}

# Trap errors and cleanup
trap 'cleanup_resources' EXIT ERR INT TERM
```

```
# Function to clean up resources
cleanup_resources() {
    local exit_code=$?

    echo "" | tee -a "$LOG_FILE"
    echo "==========================================" | tee -a "$LOG_FILE"
    echo "CLEANUP PROCESS" | tee -a "$LOG_FILE"
    echo "==========================================" | tee -a "$LOG_FILE"

    if [ -n "${DASHBOARD_NAME:-}" ]; then
        echo "Deleting CloudWatch dashboard: $DASHBOARD_NAME" | tee -a
 "$LOG_FILE"
        aws cloudwatch delete-dashboards --dashboard-names "$DASHBOARD_NAME" 2>&1
 >> "$LOG_FILE" || true
    fi

    if [ -n "${LAMBDA_FUNCTION1:-}" ]; then
        echo "Deleting Lambda function: $LAMBDA_FUNCTION1" | tee -a "$LOG_FILE"
        aws lambda delete-function --function-name "$LAMBDA_FUNCTION1" 2>&1 >>
 "$LOG_FILE" || true
    fi

    if [ -n "${LAMBDA_FUNCTION2:-}" ]; then
        echo "Deleting Lambda function: $LAMBDA_FUNCTION2" | tee -a "$LOG_FILE"
        aws lambda delete-function --function-name "$LAMBDA_FUNCTION2" 2>&1 >>
 "$LOG_FILE" || true
    fi

    if [ -n "${ROLE_NAME:-}" ]; then
        echo "Detaching policy from role: $ROLE_NAME" | tee -a "$LOG_FILE"
        aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole 2>&1 >>
 "$LOG_FILE" || true

        echo "Deleting IAM role: $ROLE_NAME" | tee -a "$LOG_FILE"
        aws iam delete-role --role-name "$ROLE_NAME" 2>&1 >> "$LOG_FILE" || true
    fi

    # Clean up temporary files securely
    shred -vfz -n 3 trust-policy.json lambda_function.py lambda_function.zip 2>/
dev/null || rm -f trust-policy.json lambda_function.py lambda_function.zip

    echo "Cleanup completed." | tee -a "$LOG_FILE"
```

```
    return $exit_code
}

# Validate AWS CLI is installed and authenticated
if ! command -v aws &> /dev/null; then
    echo "ERROR: AWS CLI is not installed" | tee -a "$LOG_FILE"
    exit 1
fi

if ! aws sts get-caller-identity &> /dev/null; then
    echo "ERROR: AWS CLI is not properly authenticated" | tee -a "$LOG_FILE"
    exit 1
fi

# Get AWS region with validation
AWS_REGION=$(aws configure get region 2>/dev/null || echo "")
if [ -z "$AWS_REGION" ]; then
    AWS_REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $AWS_REGION" | tee -a
 "$LOG_FILE"
else
    echo "Using AWS region: $AWS_REGION" | tee -a "$LOG_FILE"
fi

# Validate region format
if ! [[ "$AWS_REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]$ ]]; then
    echo "ERROR: Invalid AWS region format: $AWS_REGION" | tee -a "$LOG_FILE"
    exit 1
fi

# Generate unique identifiers using secure random with validation
RANDOM_ID=$(openssl rand -hex 6)
if [ -z "$RANDOM_ID" ] || [ ${#RANDOM_ID} -ne 12 ]; then
    echo "ERROR: Failed to generate valid random identifier" | tee -a "$LOG_FILE"
    exit 1
fi

DASHBOARD_NAME="LambdaMetricsDashboard-${RANDOM_ID}"
LAMBDA_FUNCTION1="TestFunction1-${RANDOM_ID}"
LAMBDA_FUNCTION2="TestFunction2-${RANDOM_ID}"
ROLE_NAME="LambdaExecutionRole-${RANDOM_ID}"

# Validate resource names don't exceed AWS limits
```

```
if [ ${#DASHBOARD_NAME} -gt 128 ] || [ ${#LAMBDA_FUNCTION1} -gt 64 ] ||
 [ ${#ROLE_NAME} -gt 64 ]; then
    echo "ERROR: Generated resource names exceed AWS limits" | tee -a "$LOG_FILE"
    exit 1
fi

echo "Using random identifier: $RANDOM_ID" | tee -a "$LOG_FILE"
echo "Dashboard name: $DASHBOARD_NAME" | tee -a "$LOG_FILE"
echo "Lambda function names: $LAMBDA_FUNCTION1, $LAMBDA_FUNCTION2" | tee -a
 "$LOG_FILE"
echo "IAM role name: $ROLE_NAME" | tee -a "$LOG_FILE"

# Create IAM role for Lambda functions
echo "Creating IAM role for Lambda..." | tee -a "$LOG_FILE"
TRUST_POLICY='{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}'

echo "$TRUST_POLICY" > trust-policy.json
chmod 600 trust-policy.json

# Validate JSON before use
if ! python3 -m json.tool trust-policy.json > /dev/null 2>&1; then
    echo "ERROR: Invalid trust policy JSON" | tee -a "$LOG_FILE"
    exit 1
fi

ROLE_OUTPUT=$(log_cmd "aws iam create-role --role-name '$ROLE_NAME' --assume-
role-policy-document file://trust-policy.json --tags Key=project,Value=doc-smith
 Key=tutorial,Value=cloudwatch-streams --output json")
check_error "$ROLE_OUTPUT" $? "Failed to create IAM role"

ROLE_ARN=$(echo "$ROLE_OUTPUT" | python3 -c "import sys, json;
 print(json.load(sys.stdin)['Role']['Arn'])" 2>/dev/null)
if [ -z "$ROLE_ARN" ]; then
```

```
    echo "ERROR: Failed to extract Role ARN" | tee -a "$LOG_FILE"
    exit 1
fi
echo "Role ARN: $ROLE_ARN" | tee -a "$LOG_FILE"

# Attach Lambda basic execution policy to the role
echo "Attaching Lambda execution policy to role..." | tee -a "$LOG_FILE"
POLICY_OUTPUT=$(log_cmd "aws iam attach-role-policy --role-name '$ROLE_NAME' --
policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole")
check_error "$POLICY_OUTPUT" $? "Failed to attach policy to role"

# Wait for role to propagate
echo "Waiting for IAM role to propagate..." | tee -a "$LOG_FILE"
sleep 10

# Create simple Python Lambda function code with security validation
echo "Creating Lambda function code..." | tee -a "$LOG_FILE"
cat > lambda_function.py << 'LAMBDA_EOF'
def handler(event, context):
    print("Lambda function executed successfully")
    return {
        'statusCode': 200,
        'body': 'Success'
    }
LAMBDA_EOF

chmod 600 lambda_function.py

# Validate Python syntax
if ! python3 -m py_compile lambda_function.py 2>/dev/null; then
    echo "ERROR: Invalid Python syntax in Lambda function" | tee -a "$LOG_FILE"
    exit 1
fi

# Zip the Lambda function code
zip -j -q lambda_function.zip lambda_function.py
if [ ! -f lambda_function.zip ]; then
    echo "ERROR: Failed to create lambda_function.zip" | tee -a "$LOG_FILE"
    exit 1
fi
chmod 600 lambda_function.zip

# Validate zip file integrity
if ! unzip -t lambda_function.zip > /dev/null 2>&1; then
```

```
    echo "ERROR: Created zip file is corrupted" | tee -a "$LOG_FILE"
    exit 1
fi

# Create first Lambda function
echo "Creating first Lambda function: $LAMBDA_FUNCTION1..." | tee -a "$LOG_FILE"
LAMBDA1_OUTPUT=$(log_cmd "aws lambda create-function --function-name
 '$LAMBDA_FUNCTION1' --runtime python3.11 --role '$ROLE_ARN' --handler
 lambda_function.handler --zip-file fileb://lambda_function.zip --timeout 30 --
memory-size 128 --tags project=doc-smith,tutorial=cloudwatch-streams")
check_error "$LAMBDA1_OUTPUT" $? "Failed to create first Lambda function"

# Create second Lambda function
echo "Creating second Lambda function: $LAMBDA_FUNCTION2..." | tee -a "$LOG_FILE"
LAMBDA2_OUTPUT=$(log_cmd "aws lambda create-function --function-name
 '$LAMBDA_FUNCTION2' --runtime python3.11 --role '$ROLE_ARN' --handler
 lambda_function.handler --zip-file fileb://lambda_function.zip --timeout 30 --
memory-size 128 --tags project=doc-smith,tutorial=cloudwatch-streams")
check_error "$LAMBDA2_OUTPUT" $? "Failed to create second Lambda function"

# Invoke Lambda functions to generate some metrics
echo "Invoking Lambda functions to generate metrics..." | tee -a "$LOG_FILE"
log_cmd "aws lambda invoke --function-name '$LAMBDA_FUNCTION1' --payload '{}' /
dev/null" || true
log_cmd "aws lambda invoke --function-name '$LAMBDA_FUNCTION2' --payload '{}' /
dev/null" || true

# Create CloudWatch dashboard with property variable
echo "Creating CloudWatch dashboard with property variable..." | tee -a
 "$LOG_FILE"

# Create dashboard body with proper escaping and validation
DASHBOARD_BODY=$(cat <<EOF
{
  "widgets": [
    {
      "type": "metric",
      "x": 0,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Invocations", "FunctionName", "$LAMBDA_FUNCTION1" ]
```

```
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$AWS_REGION",
        "title": "Lambda Invocations",
        "period": 300,
        "stat": "Sum"
      }
    }
  ]
}
EOF
)

# Validate JSON before sending
if ! echo "$DASHBOARD_BODY" | python3 -m json.tool > /dev/null 2>&1; then
    echo "ERROR: Dashboard body is not valid JSON" | tee -a "$LOG_FILE"
    exit 1
fi

# First create a basic dashboard without variables
echo "Creating initial dashboard without variables..." | tee -a "$LOG_FILE"
DASHBOARD_OUTPUT=$(aws cloudwatch put-dashboard --dashboard-name
 "$DASHBOARD_NAME" --dashboard-body "$DASHBOARD_BODY" --output json 2>&1)
check_error "$DASHBOARD_OUTPUT" $? "Failed to create initial CloudWatch
 dashboard"

# Now let's try to add a property variable using the console instructions
echo "To complete the tutorial, please follow these steps in the CloudWatch
 console:" | tee -a "$LOG_FILE"
echo "1. Open the CloudWatch console at https://console.aws.amazon.com/
cloudwatch/" | tee -a "$LOG_FILE"
echo "2. Navigate to Dashboards and select your dashboard: $DASHBOARD_NAME" | tee
 -a "$LOG_FILE"
echo "3. Choose Actions > Variables > Create a variable" | tee -a "$LOG_FILE"
echo "4. Choose Property variable" | tee -a "$LOG_FILE"
echo "5. For Property that the variable changes, choose FunctionName" | tee -a
 "$LOG_FILE"
echo "6. For Input type, choose Select menu (dropdown)" | tee -a "$LOG_FILE"
echo "7. Choose Use the results of a metric search" | tee -a "$LOG_FILE"
echo "8. Choose Pre-built queries > Lambda > Errors" | tee -a "$LOG_FILE"
echo "9. Choose By Function Name and then choose Search" | tee -a "$LOG_FILE"
echo "10. (Optional) Configure any secondary settings as desired" | tee -a
 "$LOG_FILE"
```

```
echo "11. Choose Add variable" | tee -a "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"
echo "The dashboard has been created and can be accessed at:" | tee -a
 "$LOG_FILE"
echo "https://console.aws.amazon.com/cloudwatch/home#dashboards:name=
$DASHBOARD_NAME" | tee -a "$LOG_FILE"

# Verify dashboard creation
echo "Verifying dashboard creation..." | tee -a "$LOG_FILE"
VERIFY_OUTPUT=$(aws cloudwatch get-dashboard --dashboard-name "$DASHBOARD_NAME"
 --output json 2>&1)
check_error "$VERIFY_OUTPUT" $? "Failed to verify dashboard creation"

echo "" | tee -a "$LOG_FILE"
echo "==========================================" | tee -a "$LOG_FILE"
echo "DASHBOARD CREATED SUCCESSFULLY" | tee -a "$LOG_FILE"
echo "==========================================" | tee -a "$LOG_FILE"
echo "Dashboard Name: $DASHBOARD_NAME" | tee -a "$LOG_FILE"
echo "Lambda Functions: $LAMBDA_FUNCTION1, $LAMBDA_FUNCTION2" | tee -a
 "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"
echo "You can view your dashboard in the CloudWatch console:" | tee -a
 "$LOG_FILE"
echo "https://console.aws.amazon.com/cloudwatch/home#dashboards:name=
$DASHBOARD_NAME" | tee -a "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"

# Auto-confirm cleanup
echo "" | tee -a "$LOG_FILE"
echo "==========================================" | tee -a "$LOG_FILE"
echo "CLEANUP CONFIRMATION" | tee -a "$LOG_FILE"
echo "==========================================" | tee -a "$LOG_FILE"
echo "The following resources were created:" | tee -a "$LOG_FILE"
echo "- CloudWatch Dashboard: $DASHBOARD_NAME" | tee -a "$LOG_FILE"
echo "- Lambda Function: $LAMBDA_FUNCTION1" | tee -a "$LOG_FILE"
echo "- Lambda Function: $LAMBDA_FUNCTION2" | tee -a "$LOG_FILE"
echo "- IAM Role: $ROLE_NAME" | tee -a "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"
echo "Auto-confirming cleanup of all created resources..." | tee -a "$LOG_FILE"

echo "Script completed successfully." | tee -a "$LOG_FILE"
exit 0
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteDashboards
  - DeleteFunction
  - DeleteRole
  - DetachRolePolicy
  - GetDashboard
  - Invoke
  - PutDashboard

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Travaillez avec DynamoDB Streams et utilisez Time-to-Live AWS Command Line Interface v2

L'exemple de code suivant montre comment gérer les flux et les fonctionnalités de DynamoDB. Time-to-Live

- Créez une table avec Streams activé.
- Décrivez Streams.
- Créez une fonction Lambda pour traiter Streams.
- Activation du TTL sur une table.
- Ajoutez des éléments dotés d'attributs TTL.
- Décrivez les paramètres TTL.

Bash

AWS CLI avec le script Bash

Créez une table avec Streams activé.

```
# Create a table with DynamoDB Streams enabled
aws dynamodb create-table \
    --table-name StreamsDemo \
    --attribute-definitions \
        AttributeName=ID,AttributeType=S \
    --key-schema \
        AttributeName=ID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --stream-specification StreamEnabled=true,StreamViewType=NEW_AND_OLD_IMAGES
```

Décrivez Streams.

```
# Get information about the stream
aws dynamodb describe-table \
    --table-name StreamsDemo \
    --query "Table.StreamSpecification"

# Get the stream ARN
STREAM_ARN=$(aws dynamodb describe-table \
    --table-name StreamsDemo \
    --query "Table.LatestStreamArn" \
    --output text)

echo "Stream ARN: $STREAM_ARN"

# Describe the stream
aws dynamodbstreams describe-stream \
    --stream-arn $STREAM_ARN
```

Créez une fonction Lambda pour Streams.

```
# Step 1: Create an IAM role for the Lambda function
cat > trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
```

```
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

aws iam create-role \
    --role-name DynamoDBStreamsLambdaRole \
    --assume-role-policy-document file://trust-policy.json

# Step 2: Attach permissions to the role
aws iam attach-role-policy \
    --role-name DynamoDBStreamsLambdaRole \
    --policy-arn arn:aws:iam::aws:policy/service-role/
AWSLambdaDynamoDBExecutionRole

# Step 3: Create a Lambda function (code would be in a separate file)
echo "Lambda function creation would be done separately with appropriate code"

# Step 4: Create an event source mapping
echo "Example command to create event source mapping:"
echo "aws lambda create-event-source-mapping \\"
echo "    --function-name ProcessDynamoDBRecords \\"
echo "    --event-source $STREAM_ARN \\"
echo "    --batch-size 100 \\"
echo "    --starting-position LATEST"
```

Activation du TTL sur une table.

```
# Create a table for TTL demonstration
aws dynamodb create-table \
    --table-name TTLDemo \
    --attribute-definitions \
        AttributeName=ID,AttributeType=S \
    --key-schema \
        AttributeName=ID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST

# Wait for table to become active
aws dynamodb wait table-exists --table-name TTLDemo
```

```
# Enable TTL on the table
aws dynamodb update-time-to-live \
    --table-name TTLDemo \
    --time-to-live-specification "Enabled=true, AttributeName=ExpirationTime"
```

Ajoutez des éléments dotés d'attributs TTL.

```
# Calculate expiration time (current time + 1 day in seconds)
EXPIRATION_TIME=$(date -d "+1 day" +%s)

# Add an item with TTL attribute
aws dynamodb put-item \
    --table-name TTLDemo \
    --item '{
        "ID": {"S": "item1"},
        "Data": {"S": "This item will expire in 1 day"},
        "ExpirationTime": {"N": "'$EXPIRATION_TIME'"}
    }'

# Add an item that expires in 1 hour
EXPIRATION_TIME_HOUR=$(date -d "+1 hour" +%s)
aws dynamodb put-item \
    --table-name TTLDemo \
    --item '{
        "ID": {"S": "item2"},
        "Data": {"S": "This item will expire in 1 hour"},
        "ExpirationTime": {"N": "'$EXPIRATION_TIME_HOUR'"}
    }'
```

Décrivez les paramètres TTL.

```
# Describe TTL settings for a table
aws dynamodb describe-time-to-live \
    --table-name TTLDemo
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreateRole

- CreateTable
- DescribeTable
- DescribeTimeToLive
- PutItem
- UpdateTimeToLive

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisez l'API IAM Policy Builder à l'aide d'un AWS Kit SDK

L'exemple de code suivant illustre comment :

- Créez des politiques IAM à l'aide de l'API orientée objet.
- Utilisez l'API IAM Policy Builder avec le service IAM.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Les exemples utilisent les importations suivantes.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.policybuilder.iam.IamConditionOperator;
import software.amazon.awssdk.policybuilder.iam.IamEffect;
import software.amazon.awssdk.policybuilder.iam.IamPolicy;
import software.amazon.awssdk.policybuilder.iam.IamPolicyWriter;
import software.amazon.awssdk.policybuilder.iam.IamPrincipal;
import software.amazon.awssdk.policybuilder.iam.IamPrincipalType;
```

```
import software.amazon.awssdk.policybuilder.iam.IamResource;
import software.amazon.awssdk.policybuilder.iam.IamStatement;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyVersionResponse;
import software.amazon.awssdk.services.sts.StsClient;

import java.net.URLDecoder;
import java.nio.charset.StandardCharsets;
import java.util.Arrays;
import java.util.List;
```

Créez une politique basée sur le temps.

```
        public String timeBasedPolicyExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")
                                                .addResource(IamResource.ALL)
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.DATE_GREATER_THAN)

 .key("aws:CurrentTime")

 .value("2020-04-01T00:00:00Z"))
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.DATE_LESS_THAN)

 .key("aws:CurrentTime")

 .value("2020-06-30T23:59:59Z")))
                                .build();

                // Use an IamPolicyWriter to write out the JSON string to a more
 readable
                // format.
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true)
```

```
                                .build());
        }
```

Créez une politique comportant plusieurs conditions.

```
        public String multipleConditionsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")

 .addAction("dynamodb:BatchGetItem")
                                                .addAction("dynamodb:Query")
                                                .addAction("dynamodb:PutItem")
                                                .addAction("dynamodb:UpdateItem")
                                                .addAction("dynamodb:DeleteItem")

 .addAction("dynamodb:BatchWriteItem")

 .addResource("arn:aws:dynamodb:*:*:table/table-name")

 .addConditions(IamConditionOperator.STRING_EQUALS

 .addPrefix("ForAllValues:"),

 "dynamodb:Attributes",
                                                                List.of("column-
name1", "column-name2", "column-name3"))
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS

 .addSuffix("IfExists"))

 .key("dynamodb:Select")

 .value("SPECIFIC_ATTRIBUTES")))
                                .build();

                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Utilisez des principaux dans une politique.

```
        public String specifyPrincipalsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.DENY)
                                                .addAction("s3:*")
                                                .addPrincipal(IamPrincipal.ALL)
                                                .addResource("arn:aws:s3:::amzn-
s3-demo-bucket/*")
                                                .addResource("arn:aws:s3:::amzn-
s3-demo-bucket")
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.ARN_NOT_EQUALS)

 .key("aws:PrincipalArn")

 .value("arn:aws:iam::444455556666:user/user-name")))
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Autorisez l'accès intercompte.

```
        public String allowCrossAccountAccessExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)

 .addPrincipal(IamPrincipalType.AWS, "111122223333")
                                                .addAction("s3:PutObject")
                                                .addResource("arn:aws:s3:::amzn-
s3-demo-bucket/*")
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS)
                                                                .key("s3:x-amz-
acl")
```

```
                                                                    .value("bucket-
owner-full-control")))
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Créez et chargez une `IamPolicy`.

```
        public String createAndUploadPolicyExample(IamClient iam, String
 accountID, String policyName) {
                // Build the policy.
                IamPolicy policy = IamPolicy.builder() // 'version' defaults to
 "2012-10-17".
                                .addStatement(IamStatement.builder()
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:PutItem")

 .addResource("arn:aws:dynamodb:us-east-1:" + accountID
                                                                    + ":table/
exampleTableName")
                                                .build())
                                .build();
                // Upload the policy.
                iam.createPolicy(r ->
 r.policyName(policyName).policyDocument(policy.toJson()));
                return
 policy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
        }
```

Téléchargez et utilisez une `IamPolicy`.

```
        public String createNewBasedOnExistingPolicyExample(IamClient iam, String
 accountID, String policyName,
                        String newPolicyName) {

                String policyArn = "arn:aws:iam::" + accountID + ":policy/" +
 policyName;
                GetPolicyResponse getPolicyResponse = iam.getPolicy(r ->
 r.policyArn(policyArn));
```

```
                String policyVersion =
getPolicyResponse.policy().defaultVersionId();
                GetPolicyVersionResponse getPolicyVersionResponse = iam
                                .getPolicyVersion(r ->
r.policyArn(policyArn).versionId(policyVersion));

                // Create an IamPolicy instance from the JSON string returned
from IAM.
                String decodedPolicy =
URLDecoder.decode(getPolicyVersionResponse.policyVersion().document(),
                                StandardCharsets.UTF_8);
                IamPolicy policy = IamPolicy.fromJson(decodedPolicy);

                /*
                 * All IamPolicy components are immutable, so use the copy method
that creates a
                 * new instance that
                 * can be altered in the same method call.
                 *
                 * Add the ability to get an item from DynamoDB as an additional
action.
                 */
                IamStatement newStatement = policy.statements().get(0).copy(s ->
s.addAction("dynamodb:GetItem"));

                // Create a new statement that replaces the original statement.
                IamPolicy newPolicy = policy.copy(p ->
p.statements(Arrays.asList(newStatement)));

                // Upload the new policy. IAM now has both policies.
                iam.createPolicy(r -> r.policyName(newPolicyName)
                                .policyDocument(newPolicy.toJson()));

                return
newPolicy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
        }
```

- Pour plus d'informations, consultez le Guide du développeur AWS SDK for Java 2.x.
- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK for Java 2.x .
  - CreatePolicy

- GetPolicy
- GetPolicyVersion

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Exemples de code pour AWS STS utilisant AWS Kits SDK

Les exemples de code suivants montrent comment utiliser AWS STS un kit de développement AWS logiciel (SDK).

Les actions sont des extraits de code de programmes plus larges et doivent être exécutées dans leur contexte. Alors que les actions vous indiquent comment appeler des fonctions de service individuelles, vous pouvez les voir en contexte dans leurs scénarios associés.

Les scénarios sont des exemples de code qui vous montrent comment accomplir des tâches spécifiques en appelant plusieurs fonctions au sein d'un même service ou combinés à d'autres Services AWS.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

AWS STS

- Exemples de base pour AWS STS utilisant AWS Kits SDK
  - Actions pour AWS STS utilisant AWS Kits SDK
    - À utiliser AssumeRoleavec un AWS SDK ou CLI
    - Utilisation de AssumeRoleWithWebIdentity avec une CLI
    - Utilisation de DecodeAuthorizationMessage avec une CLI
    - Utilisation de GetFederationToken avec une CLI
    - À utiliser GetSessionTokenavec un AWS SDK ou CLI
- Scénarios pour AWS STS utilisant AWS Kits SDK
  - Assumez un rôle IAM qui nécessite un jeton MFA avec AWS STS à l'aide d'un AWS Kit SDK
  - Configuration de la connectivité des services de conteneurs

- Construisez une URL avec AWS STS pour les utilisateurs fédérés utilisant un AWS Kit SDK
- Création d'une tâche de conteneur pour le type de lancement sans serveur
- Création d'une API REST avec intégration de proxy de fonctions
- Création d'un service de conteneur pour les instances de machines virtuelles
- Création d'un espace de travail de surveillance géré
- Création d'un tableau de bord de surveillance avec le nom de la fonction comme variable
- Obtenez un jeton de session qui nécessite un jeton MFA avec AWS STS à l'aide d'un AWS Kit SDK
- Commencer à gérer la configuration
- Commencer à utiliser les registres de conteneurs
- Débuter avec la messagerie de l'Internet des objets
- Commencer à utiliser les magasins de fonctionnalités d'apprentissage automatique
- Commencer à utiliser les clusters Kubernetes gérés
- Débuter avec le streaming géré
- Commencer à utiliser les notifications push
- Commencer à utiliser les moteurs de recherche et d'analyse
- Commencer à utiliser les pare-feux pour applications Web
- Transférer les secrets codés en dur vers un stockage secret sécurisé
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts
- Configuration de la gestion des systèmes

Exemples de base pour AWS STS utilisant AWS Kits SDK

Les exemples de code suivants montrent comment utiliser les principes de base AWS Security Token Service des AWS SDK.

Exemples

- Actions pour AWS STS utilisant AWS Kits SDK
  - À utiliser AssumeRoleavec un AWS SDK ou CLI
  - Utilisation de AssumeRoleWithWebIdentity avec une CLI
  - Utilisation de DecodeAuthorizationMessage avec une CLI

- Utilisation de GetFederationToken avec une CLI
- À utiliser GetSessionTokenavec un AWS SDK ou CLI

Actions pour AWS STS utilisant AWS Kits SDK

Les exemples de code suivants montrent comment effectuer des AWS STS actions individuelles avec AWS les SDK. Chaque exemple inclut un lien vers GitHub, où vous pouvez trouver des instructions pour configurer et exécuter le code.

Ces extraits appellent l' AWS STS API et sont des extraits de code de programmes plus volumineux qui doivent être exécutés en contexte. Vous pouvez voir les actions dans leur contexte dans Scénarios pour AWS STS utilisant AWS Kits SDK.

Les exemples suivants incluent uniquement les actions les plus couramment utilisées. Pour obtenir la liste complète, consultez la Référence des API AWS Security Token Service.

Exemples

- À utiliser AssumeRoleavec un AWS SDK ou CLI
- Utilisation de AssumeRoleWithWebIdentity avec une CLI
- Utilisation de DecodeAuthorizationMessage avec une CLI
- Utilisation de GetFederationToken avec une CLI
- À utiliser GetSessionTokenavec un AWS SDK ou CLI

À utiliser **`AssumeRole`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `AssumeRole`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Assumer un rôle IAM qui nécessite un jeton MFA
- Créer une URL pour utilisateurs fédérés

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;

namespace AssumeRoleExample
{
    class AssumeRole
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
        ///
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        // A region property may be used if the profile or credentials loaded do
 not specify a region,
        // or to use a specific region.
        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
```

```
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"Original Caller: {caller.Arn}");

            // Create the request to use with the AssumeRoleAsync call.
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);

            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials, REGION);

            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK pour .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
```

```
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
```

```
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
                             const Aws::String &externalId,
                             Aws::Auth::AWSCredentials &credentials,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;

    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
```

```
                    outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour endosser un rôle

La commande `assume-role` suivante extrait un ensemble d'informations d'identification à court terme pour le rôle IAM `s3-access-example`.

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

Sortie :

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
```

```
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
    },
    "Credentials": {
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

La sortie de la commande contient une clé d'accès, une clé secrète et un jeton de session que vous pouvez utiliser pour vous authentifier auprès d' AWS.

Pour l'utilisation de la AWS CLI, vous pouvez configurer un profil nommé associé à un rôle. Lorsque vous utilisez le profil, la AWS CLI appelle assume-role et gère les informations d'identification pour vous. Pour plus d'informations, consultez la section Utiliser un rôle IAM dans la AWS CLI dans le Guide de l'utilisateur de la AWS CLI.

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
```

```
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
import java.time.Instant;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
 * example:
 *
 * {
 * "Version":"2012-10-17",
 * "Statement": [
 * {
 * "Effect": "Allow",
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code example>"
 * },
 * "Action": "sts:AssumeRole"
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
 * Role" in the AWS Directory Service guide.
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s
```

```
                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
assume (for example, arn:aws:iam::000008047983:role/s3role).\s
                    roleSessionName - An identifier for the assumed role session
(for example, mysession).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
                .region(region)
                .build();

        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }

    public static void assumeGivenRole(StsClient stsClient, String roleArn,
String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());
```

```
            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
 exTime);

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez le client.

```
import { STSClient } from "@aws-sdk/client-sts";
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

Assumez le rôle IAM.

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
```

```
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK pour JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
```

```
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });

//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
});

//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);

  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK pour JavaScript API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : renvoie un ensemble d'informations d'identification temporaires (clé d'accès, clé secrète et jeton de session) qui peuvent être utilisées pendant une heure pour accéder

à AWS des ressources auxquelles l'utilisateur demandeur n'a pas normalement accès. Les informations d'identification renvoyées disposent des autorisations autorisées par la stratégie d'accès du rôle assumé et par la politique fournie (vous ne pouvez pas utiliser la politique fournie pour accorder des autorisations supérieures à celles définies par la stratégie d'accès du rôle assumé).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Exemple 2 : renvoie un ensemble d'informations d'identification temporaires, valides pendant une heure, dotées des mêmes autorisations que celles définies dans la stratégie d'accès du rôle assumé.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Exemple 3 : renvoie un ensemble d'informations d'identification temporaires fournissant le numéro de série et le jeton généré par une MFA associée aux informations d'identification de l'utilisateur utilisées pour exécuter l'applet de commande.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Exemple 4 : renvoie un ensemble d'informations d'identification temporaires qui ont endossé un rôle défini dans le compte d'un client. Pour chaque rôle que le tiers peut assumer, le compte client doit créer un rôle à l'aide d'un identifiant qui doit être transmis dans le ExternalId paramètre - chaque fois que le rôle est assumé.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : renvoie un ensemble d'informations d'identification temporaires (clé d'accès, clé secrète et jeton de session) qui peuvent être utilisées pendant une heure pour accéder

à AWS des ressources auxquelles l'utilisateur demandeur n'a pas normalement accès. Les informations d'identification renvoyées disposent des autorisations autorisées par la stratégie d'accès du rôle assumé et par la politique fournie (vous ne pouvez pas utiliser la politique fournie pour accorder des autorisations supérieures à celles définies par la stratégie d'accès du rôle assumé).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Exemple 2 : renvoie un ensemble d'informations d'identification temporaires, valides pendant une heure, dotées des mêmes autorisations que celles définies dans la stratégie d'accès du rôle assumé.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Exemple 3 : renvoie un ensemble d'informations d'identification temporaires fournissant le numéro de série et le jeton généré par une MFA associée aux informations d'identification de l'utilisateur utilisées pour exécuter l'applet de commande.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Exemple 4 : renvoie un ensemble d'informations d'identification temporaires qui ont endossé un rôle défini dans le compte d'un client. Pour chaque rôle que le tiers peut assumer, le compte client doit créer un rôle à l'aide d'un identifiant qui doit être transmis dans le ExternalId paramètre - chaque fois que le rôle est assumé.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Assumez un rôle IAM qui nécessite un jeton MFA et utilisez des informations d'identification temporaires pour répertorier les compartiments Amazon S3 pour le compte.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")
```

```
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Pour plus de détails sur l'API, consultez AssumeRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
```

```
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: 'create-use-assume-role-scenario'
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
```

```
        .load()
        .await;
    let client = Client::new(&local_config);
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
        Err(e) => println!("{:?}", e),
    }
}
```

- Pour plus de détails sur l'API, voir AssumeRolela section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSSTS

    public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
        async throws -> STSClientTypes.Credentials
    {
        let input = AssumeRoleInput(
            roleArn: role.arn,
            roleSessionName: sessionName
        )
        do {
```

```
            let output = try await stsClient.assumeRole(input: input)

            guard let credentials = output.credentials else {
                throw ServiceHandlerError.authError
            }

            return credentials
        } catch {
            print("Error assuming role: ", dump(error))
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`AssumeRoleWithWebIdentity`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `AssumeRoleWithWebIdentity`.

CLI

AWS CLI

Pour obtenir des informations d'identification à court terme pour un rôle authentifié avec Web Identity (OAuth 2.0)

La commande `assume-role-with-web-identity` suivante extrait un ensemble d'informations d'identification à court terme pour le rôle IAM `app1`. La requête est authentifiée à l'aide du jeton d'identité Web fourni par le fournisseur d'identité Web spécifié. Deux politiques supplémentaires sont appliquées à la session afin de restreindre davantage ce que l'utilisateur peut faire. Les informations d'identification renvoyées expirent une heure après avoir été générées.

```
aws sts assume-role-with-web-identity \
    --duration-seconds 3600 \
```

```
    --role-session-name "app1" \
    --provider-id "www.amazon.com" \
    --policy-arns "arn:aws:iam::123456789012:policy/
q=webidentitydemopolicy1","arn:aws:iam::123456789012:policy/
webidentitydemopolicy2" \
    --role-arn arn:aws:iam::123456789012:role/FederatedWebIdentityRole \
    --web-identity-token "Atza
%7CIQEBLjAsAhRFiXuWpUXuRvQ9PZL3GMFcYevydwIUFAHZwXZXXXXXXXXJnrulxKDHwy87oGKPznh0D6bEQZTSCz
CrKqjG7nPBjNIL016GGvuS5gSvPRUxWES3VYfm1wl7WTI7jn-Pcb6M-
buCgHhFOzTQxod27L9CqnOLio7N3gZAGpsp6n1-
AJBOCJckcyXe2c6uD0srOJeZlKUm2eTDVMf8IehDVI0r1QOnTV6KzzAI3OY87Vd_cVMQ"
```

Sortie :

```
{
    "SubjectFromWebIdentityToken": "amzn1.account.AF6RHO7KZU5XRVQJGXK6HB56KR2A",
    "Audience": "client.5498841531868486423.1548@apps.example.com",
    "AssumedRoleUser": {
        "Arn": "arn:aws:sts::123456789012:assumed-role/FederatedWebIdentityRole/
app1",
        "AssumedRoleId": "AROACLKWSDQRAOEXAMPLE:app1"
    },
    "Credentials": {
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    },
    "Provider": "www.amazon.com"
}
```

Pour plus d'informations, consultez Demande d'informations d'identification temporaires de sécurité dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous AssumeRoleWithWebIdentityà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : renvoie un ensemble d'informations d'identification temporaires, valables pendant une heure, pour un utilisateur qui a été authentifié avec le fournisseur d'identité Login with Amazon. Les informations d'identification reposent sur la stratégie d'accès associée au rôle identifié par l'ARN du rôle. En option, vous pouvez transmettre une politique JSON au paramètre -Policy qui affine davantage les autorisations d'accès (vous ne pouvez pas accorder plus d'autorisations que celles qui sont disponibles dans les autorisations associées au rôle). La valeur fournie à - WebIdentityToken est l'identifiant utilisateur unique renvoyé par le fournisseur d'identité.

```
Use-STSWebIdentityRole -DurationInSeconds 3600 -ProviderId "www.amazon.com"
 -RoleSessionName "app1" -RoleArn "arn:aws:iam::123456789012:role/
FederatedWebIdentityRole" -WebIdentityToken "Atza...DVI0r1"
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleWithWebIdentityà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : renvoie un ensemble d'informations d'identification temporaires, valables pendant une heure, pour un utilisateur qui a été authentifié avec le fournisseur d'identité Login with Amazon. Les informations d'identification reposent sur la stratégie d'accès associée au rôle identifié par l'ARN du rôle. En option, vous pouvez transmettre une politique JSON au paramètre -Policy qui affine davantage les autorisations d'accès (vous ne pouvez pas accorder plus d'autorisations que celles qui sont disponibles dans les autorisations associées au rôle). La valeur fournie à - WebIdentityToken est l'identifiant utilisateur unique renvoyé par le fournisseur d'identité.

```
Use-STSWebIdentityRole -DurationInSeconds 3600 -ProviderId "www.amazon.com"
 -RoleSessionName "app1" -RoleArn "arn:aws:iam::123456789012:role/
FederatedWebIdentityRole" -WebIdentityToken "Atza...DVI0r1"
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleWithWebIdentityà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`DecodeAuthorizationMessage`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `DecodeAuthorizationMessage`.

CLI

AWS CLI

Pour décoder un message d'autorisation codé renvoyé en réponse à une requête

L'exemple `decode-authorization-message` suivant décode des informations supplémentaires sur le statut d'autorisation d'une demande émise depuis un message codé envoyé en réponse à une requête Amazon Web Services.

```
aws sts decode-authorization-message \
    --encoded-message EXAMPLEWodyRNrtlQARDip-
eTA6i6DrlUhHhPQrLWB_lAbl5pAKxl9mPDLexYcGBreyIKQC1BGBIpBKr3dFDkwqeO7e2NMk5j_hmzAiChJN-8oy3
Ojau7BMjOTWwOtHPhV_Zaz87yENdipr745EjQwRd5LaoL3vN8_5ZfA9UiBMKDgVh1gjqZJFUiQoubv78V1RbHNYnK
pOu3FZjwYStfvTb3GHs3-6rLribGO9jZOktkfE6vqxlFzLyeDr4P2ihC1wty9tArCvvGzIAUNmARQJ2VVWPxioqgo
JWP5pwe_mAyqh0NLw-r1S56YC_90onj9A80sNrHlI-
tIiNd7tgNTYzDuPQYD2FMDBnp82V9eVmYGtPp5NIeSpuf3fOHanFuBZgENxZQZ2dlH3xJGMTtYayzZrRXjiq_SfX9
FaoPIb8LmmKVBLpIB0iFhU9sEHPqKHVPi6jdxXqKaZaFGvYVmVOiuQdNQKuyk0p067POFrZECLjjOtNPBOZCcuEKE
```

Sortie :

```
{
    "DecodedMessage": "{\"allowed\":false,\"explicitDeny\":true,
\"matchedStatements\":{\"items\":[{\"statementId\":\"VisualEditor0\",\"effect
\":\"DENY\",\"principals\":{\"items\":[{\"value\":\"AROA123456789EXAMPLE
\"}]},\"principalGroups\":{\"items\":[]},\"actions\":{\"items\":[{\"value
\":\"ec2:RunInstances\"}]},\"resources\":{\"items\":[{\"value\":\"*
\"}]},\"conditions\":{\"items\":[]}}]},\"failures\":{\"items\":[]},
\"context\":{\"principal\":{\"id\":\"AROA123456789EXAMPLE:Ana\",\"arn
\":\"arn:aws:sts::111122223333:assumed-role/Developer/Ana\"},\"action\":
\"RunInstances\",\"resource\":\"arn:aws:ec2:us-east-1:111122223333:instance/*
\",\"conditions\":{\"items\":[{\"key\":\"ec2:MetadataHttpPutResponseHopLimit\",
\"values\":{\"items\":[{\"value\":\"2\"}]}},{\"key\":\"ec2:InstanceMarketType
\",\"values\":{\"items\":[{\"value\":\"on-demand\"}]}},{\"key\":\"aws:Resource
```

```
\",\"values\":{\"items\":[{\"value\":\"instance/*\"}]}},{\"key\":\"aws:Account
\",\"values\":{\"items\":[{\"value\":\"111122223333\"}]}},{\"key\":
\"ec2:AvailabilityZone\",\"values\":{\"items\":[{\"value\":\"us-east-1f\"}]}},
{\"key\":\"ec2:ebsOptimized\",\"values\":{\"items\":[{\"value\":\"false\"}]}},
{\"key\":\"ec2:IsLaunchTemplateResource\",\"values\":{\"items\":[{\"value\":
\"false\"}]}},{\"key\":\"ec2:InstanceType\",\"values\":{\"items\":[{\"value
\":\"t2.micro\"}]}},{\"key\":\"ec2:RootDeviceType\",\"values\":{\"items\":
[{\"value\":\"ebs\"}]}},{\"key\":\"aws:Region\",\"values\":{\"items\":[{\"value
\":\"us-east-1\"}]}},{\"key\":\"ec2:MetadataHttpEndpoint\",\"values\":{\"items
\":[{\"value\":\"enabled\"}]}},{\"key\":\"aws:Service\",\"values\":{\"items
\":[{\"value\":\"ec2\"}]}},{\"key\":\"ec2:InstanceID\",\"values\":{\"items\":
[{\"value\":\"*\"}]}},{\"key\":\"ec2:MetadataHttpTokens\",\"values\":{\"items
\":[{\"value\":\"required\"}]}},{\"key\":\"aws:Type\",\"values\":{\"items
\":[{\"value\":\"instance\"}]}},{\"key\":\"ec2:Tenancy\",\"values\":{\"items
\":[{\"value\":\"default\"}]}},{\"key\":\"ec2:Region\",\"values\":{\"items
\":[{\"value\":\"us-east-1\"}]}},{\"key\":\"aws:ARN\",\"values\":{\"items\":
[{\"value\":\"arn:aws:ec2:us-east-1:111122223333:instance/*\"}]}}]}}"
}
```

Pour plus d'informations, consultez Logique d'évaluation des politiques dans le AWS Guide de l'utilisateur IAM.

- Pour plus de détails sur l'API, reportez-vous DecodeAuthorizationMessageà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : décode les informations supplémentaires contenues dans le contenu du message codé fourni qui a été renvoyé en réponse à une demande. Les informations supplémentaires sont codées parce que les détails du statut de l'autorisation peuvent constituer des informations privilégiées que l'utilisateur qui a demandé l'action ne devrait pas voir.

```
Convert-STSAuthorizationMessage -EncodedMessage "...encoded message..."
```

- Pour plus de détails sur l'API, reportez-vous DecodeAuthorizationMessageà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : décode les informations supplémentaires contenues dans le contenu du message codé fourni qui a été renvoyé en réponse à une demande. Les informations supplémentaires sont codées parce que les détails du statut de l'autorisation peuvent constituer des informations privilégiées que l'utilisateur qui a demandé l'action ne devrait pas voir.

```
Convert-STSAuthorizationMessage -EncodedMessage "...encoded message..."
```

- Pour plus de détails sur l'API, reportez-vous DecodeAuthorizationMessageà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation de **`GetFederationToken`** avec une CLI

Les exemples de code suivants illustrent comment utiliser `GetFederationToken`.

CLI

AWS CLI

Pour renvoyer un ensemble d'informations d'identification de sécurité temporaires à l'aide des informations d'identification de la clé d'accès utilisateur IAM

L'exemple `get-federation-token` suivant renvoie un ensemble d'informations d'identification de sécurité temporaires (composé d'un ID de clé d'accès, d'une clé d'accès secrète et d'un jeton de sécurité) pour un utilisateur. Vous devez appeler l'opération `GetFederationToken` à l'aide des informations d'identification de sécurité à long terme d'un utilisateur IAM.

```
aws sts get-federation-token \
    --name Bob \
    --policy file://myfile.json \
    --policy-arns arn=arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess \
    --duration-seconds 900
```

Contenu de `myfile.json` :

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:ListMetrics",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:Describe*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "autoscaling:Describe*",
            "Resource": "*"
        }
    ]
}
```

Sortie :

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "SessionToken": "EXAMPLEpZ2luX2VjEGoaCXVzLXdlc3QtMiJIMEYCIQC/
W9pL5ArQyDD5JwFL3/h5+WGopQ24GEXweNctwhi9sgIhAMkg
+MZE35iWM8s4r5Lr25f9rSTVPFH98G42QQunWMTfKq0DCOP/////////
wEQAxoMNDUyOTI1MTcwNTA3Igxuy3AOpuuoLsk3MJwqgQPg8QOd9HuoClUxq26wnc/nm
+eZLjHDyGf2KUAHK2DuaS/nrGSEXAMPLE",
```

```
        "Expiration": "2023-12-20T02:06:07+00:00"
    },
    "FederatedUser": {
        "FederatedUserId": "111122223333:Bob",
        "Arn": "arn:aws:sts::111122223333:federated-user/Bob"
    },
    "PackedPolicySize": 36
}
```

Pour plus d'informations, consultez Demande d'informations d'identification temporaires de sécurité dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetFederationTokenà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : demande un jeton fédéré valable une heure en utilisant « Bob » comme nom de l'utilisateur fédéré. Ce nom peut être utilisé pour référencer le nom de l'utilisateur fédéré dans une politique basée sur les ressources (telle qu'une politique de compartiment Amazon S3). La politique IAM fournie, au format JSON, est utilisée pour limiter les autorisations disponibles pour l'utilisateur IAM. La politique fournie ne peut accorder plus d'autorisations que celles accordées à l'utilisateur qui effectue la demande, les autorisations finales pour l'utilisateur fédéré constituant l'ensemble le plus restrictif à la croisée de la politique transmise et la politique de l'utilisateur IAM.

```
Get-STSFederationToken -Name "Bob" -Policy "...JSON policy..." -DurationInSeconds
 3600
```

- Pour plus de détails sur l'API, reportez-vous GetFederationTokenà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : demande un jeton fédéré valable une heure en utilisant « Bob » comme nom de l'utilisateur fédéré. Ce nom peut être utilisé pour référencer le nom de l'utilisateur fédéré dans une politique basée sur les ressources (telle qu'une politique de compartiment Amazon S3). La politique IAM fournie, au format JSON, est utilisée pour limiter les autorisations disponibles pour l'utilisateur IAM. La politique fournie ne peut accorder plus d'autorisations que celles

accordées à l'utilisateur qui effectue la demande, les autorisations finales pour l'utilisateur fédéré constituant l'ensemble le plus restrictif à la croisée de la politique transmise et la politique de l'utilisateur IAM.

```
Get-STSFederationToken -Name "Bob" -Policy "...JSON policy..." -DurationInSeconds
 3600
```

- Pour plus de détails sur l'API, reportez-vous GetFederationTokenà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

À utiliser **`GetSessionToken`**avec un AWS SDK ou CLI

Les exemples de code suivants illustrent comment utiliser `GetSessionToken`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Obtenir un jeton de session qui nécessite un jeton MFA

CLI

AWS CLI

Pour obtenir un ensemble d'informations d'identification à court terme d'une identité IAM

La commande `get-session-token` suivante extrait un ensemble d'informations d'identification à court terme pour l'identité IAM qui effectue l'appel. Les informations d'identification obtenues peuvent être utilisées pour les requêtes où l'authentification multifactorielle (MFA) est requise par la politique. Les informations d'identification expirent 15 minutes après leur création.

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

Sortie :

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

Pour plus d'informations, consultez Demande d'informations d'identification temporaires de sécurité dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetSessionTokenà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell V4

Exemple 1 : renvoie une instance **`Amazon.RuntimeAWSCredentials`** contenant des informations d'identification temporaires valables pendant une période définie. Les informations d'identification utilisées pour demander des informations d'identification temporaires sont déduites des paramètres par défaut actuels du shell. Pour spécifier d'autres informations d'identification, utilisez les SecretKey paramètres - ProfileName ou - AccessKey /-.

```
Get-STSSessionToken
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemple 2 : renvoie une instance **Amazon.RuntimeAWSCredentials** contenant des informations d'identification temporaires valides pour une heure. Les informations d'identification utilisées pour effectuer la demande sont obtenues à partir du profil spécifié.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemple 3 : renvoie une instance **Amazon.RuntimeAWSCredentials** contenant des informations d'identification temporaires valables pendant une heure en utilisant le numéro d'identification du dispositif MFA associé au compte dont les informations d'identification sont spécifiées dans le profil « myprofilename » et la valeur fournie par le dispositif.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- Pour plus de détails sur l'API, reportez-vous GetSessionTokenà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : renvoie une instance **Amazon.RuntimeAWSCredentials** contenant des informations d'identification temporaires valables pendant une période définie. Les informations d'identification utilisées pour demander des informations d'identification

temporaires sont déduites des paramètres par défaut actuels du shell. Pour spécifier d'autres informations d'identification, utilisez les SecretKey paramètres - ProfileName ou - AccessKey /-.

```
Get-STSSessionToken
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemple 2 : renvoie une instance **Amazon.RuntimeAWSCredentials** contenant des informations d'identification temporaires valides pour une heure. Les informations d'identification utilisées pour effectuer la demande sont obtenues à partir du profil spécifié.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemple 3 : renvoie une instance **Amazon.RuntimeAWSCredentials** contenant des informations d'identification temporaires valables pendant une heure en utilisant le numéro d'identification du dispositif MFA associé au compte dont les informations d'identification sont spécifiées dans le profil « myprofilename » et la valeur fournie par le dispositif.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Sortie :

```
AccessKeyId                               Expiration
 SecretAccessKey                          SessionToken
-----------                               ----------
 ---------------                          ------------
EXAMPLEACCESSKEYID                        2/16/2015 9:12:28 PM
 examplesecretaccesskey...                SamPleTokeN.....
```

- Pour plus de détails sur l'API, reportez-vous GetSessionTokenà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Obtenez un jeton de session en transmettant un jeton MFA et utilisez-le pour répertorier les compartiments Amazon S3 pour le compte.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
```

```
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Pour plus de détails sur l'API, consultez GetSessionTokenle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Scénarios pour AWS STS utilisant AWS Kits SDK

Les exemples de code suivants vous montrent comment implémenter des scénarios courants AWS STS avec AWS les SDK. Ces scénarios vous montrent comment accomplir des tâches spécifiques en appelant plusieurs fonctions AWS STS ou en les combinant avec d'autres Services AWS. Chaque exemple inclut un lien vers le code source complet, où vous trouverez des instructions sur la configuration et l'exécution du code.

Les scénarios ciblent un niveau d'expérience intermédiaire pour vous aider à comprendre les actions de service dans leur contexte.

Exemples

- Assumez un rôle IAM qui nécessite un jeton MFA avec AWS STS à l'aide d'un AWS Kit SDK
- Configuration de la connectivité des services de conteneurs

- Construisez une URL avec AWS STS pour les utilisateurs fédérés utilisant un AWS Kit SDK
- Création d'une tâche de conteneur pour le type de lancement sans serveur
- Création d'une API REST avec intégration de proxy de fonctions
- Création d'un service de conteneur pour les instances de machines virtuelles
- Création d'un espace de travail de surveillance géré
- Création d'un tableau de bord de surveillance avec le nom de la fonction comme variable
- Obtenez un jeton de session qui nécessite un jeton MFA avec AWS STS à l'aide d'un AWS Kit SDK
- Commencer à gérer la configuration
- Commencer à utiliser les registres de conteneurs
- Débuter avec la messagerie de l'Internet des objets
- Commencer à utiliser les magasins de fonctionnalités d'apprentissage automatique
- Commencer à utiliser les clusters Kubernetes gérés
- Débuter avec le streaming géré
- Commencer à utiliser les notifications push
- Commencer à utiliser les moteurs de recherche et d'analyse
- Commencer à utiliser les pare-feux pour applications Web
- Transférer les secrets codés en dur vers un stockage secret sécurisé
- Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts
- Configuration de la gestion des systèmes

Assumez un rôle IAM qui nécessite un jeton MFA avec AWS STS à l'aide d'un AWS Kit SDK

L'exemple de code suivant montre comment assumer un rôle qui nécessite un jeton MFA.

> **Warning**
>
> Afin d'éviter les risques de sécurité, n'employez pas les utilisateurs IAM pour l'authentification lorsque vous développez des logiciels spécialisés ou lorsque vous travaillez avec des

données réelles. Préférez la fédération avec un fournisseur d'identité tel que AWS IAM Identity Center.

- Créez un rôle IAM qui accorde l'autorisation de répertorier les compartiments Amazon S3.
- Créez un utilisateur IAM autorisé à assumer le rôle uniquement lorsque des informations d'identification MFA sont fournies.
- Enregistrez un périphérique MFA pour l'utilisateur.
- Assumer le rôle et utiliser des informations d'identification temporaires pour répertorier des compartiments S3.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez un utilisateur IAM, enregistrez un dispositif MFA et créez un rôle qui accorde l'autorisation de répertorier des compartiments S3. L'utilisateur n'a que le droit d'assumer le rôle.

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual MFA device.
    Displays the QR code to seed the device.
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role and requires
 MFA.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.
```

```
    For demonstration purposes, the user is created in the same account as the
role,
    but in practice the user would likely be from another account.

    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
resource
                         that has permissions to create users, roles, and
policies
                         in the account.
    :return: The newly created user, user key, virtual MFA device, and role.
    """
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")

    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")

    user_key = user.create_access_key_pair()
    print(f"Created access key pair for user.")
```

```
    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": user.arn},
                        "Action": "sts:AssumeRole",
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
True}},
                    }
                ],
            }
        ),
    )
    print(f"Created role {role.name} that requires MFA.")

    policy = iam_resource.create_policy(
        PolicyName=unique_name("policy"),
        PolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
                    }
                ],
            }
        ),
    )
    role.attach_policy(PolicyArn=policy.arn)
    print(f"Created policy {policy.policy_name} and attached it to the role.")

    user.create_policy(
        PolicyName=unique_name("user-policy"),
        PolicyDocument=json.dumps(
```

```
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "sts:AssumeRole",
                        "Resource": role.arn,
                    }
                ],
            }
        ),
    )
    print(
        f"Created an inline policy for {user.name} that lets the user assume "
        f"the role."
    )

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device, role
```

Montrez qu'assumer le rôle sans jeton MFA n'est pas autorisé.

```
def try_to_assume_role_without_mfa(assume_role_arn, session_name, sts_client):
    """
    Shows that attempting to assume the role without sending MFA credentials
 results
    in an AccessDenied error.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role to assume.
    :param session_name: The name of the STS session.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    print(f"Trying to assume the role without sending MFA credentials...")
    try:
        sts_client.assume_role(RoleArn=assume_role_arn,
 RoleSessionName=session_name)
```

```
            raise RuntimeError("Expected AccessDenied error.")
        except ClientError as error:
            if error.response["Error"]["Code"] == "AccessDenied":
                print("Got AccessDenied.")
            else:
                raise
```

Assumez le rôle qui accorde l'autorisation de répertorier des compartiments S3, en transmettant le jeton MFA requis, et montrez que les compartiments peuvent être répertoriés.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")
```

```
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

Détruisez les ressources créées pour la démo.

```
def teardown(user, virtual_mfa_device, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    for attached in role.attached_policies.all():
        policy_name = attached.policy_name
        role.detach_policy(PolicyArn=attached.arn)
        attached.delete()
        print(f"Detached and deleted {policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
        key.delete()
        print("Deleted user's access key.")
    for mfa in user.mfa_devices.all():
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

Exécutez ce scénario à l'aide des fonctions définies précédemment.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(
        f"Welcome to the AWS Security Token Service assume role demo, "
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device, role = setup(iam_resource)
    print(f"Created {user.name} and {role.name}.")
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
        )
        try_to_assume_role_without_mfa(role.arn, "demo-sts-session", sts_client)
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_from_assumed_role_with_mfa(
            role.arn,
            "demo-sts-session",
            virtual_mfa_device.serial_number,
            mfa_totp,
            sts_client,
        )
    finally:
        teardown(user, virtual_mfa_device, role)
        print("Thanks for watching!")
```

- Pour plus de détails sur l'API, consultez AssumeRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Configuration de la connectivité des services de conteneurs

L'exemple de code suivant illustre comment :

- Création de l'infrastructure VPC
- Configuration de la journalisation
- Création du cluster ECS
- Configuration des rôles IAM
- Créez le service avec Service Connect
- Vérification du déploiement
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# ECS Service Connect Tutorial Script v4 - Modified to use Default VPC
# This script creates an ECS cluster with Service Connect and deploys an nginx
 service
# Uses the default VPC to avoid VPC limits

set -e  # Exit on any error

# Configuration
SCRIPT_NAME="ECS Service Connect Tutorial"
LOG_FILE="ecs-service-connect-tutorial-v4-default-vpc.log"
REGION=${AWS_DEFAULT_REGION:-${AWS_REGION:-$(aws configure get region 2>/dev/
null)}}
if [ -z "$REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Set one with: aws configure set region us-east-1"
```

```
    exit 1
fi
ENV_PREFIX="tutorial"
CLUSTER_NAME="${ENV_PREFIX}-cluster"
NAMESPACE_NAME="service-connect"

# Generate random suffix for unique resource names
RANDOM_SUFFIX=$(openssl rand -hex 6)

# Arrays to track created resources for cleanup
declare -a CREATED_RESOURCES=()

# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
}

# Error handling function
handle_error() {
    log "ERROR: Script failed at line $1"
    log "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set up error handling
trap 'handle_error $LINENO' ERR

# Function to add resource to tracking array
track_resource() {
    CREATED_RESOURCES+=("$1")
    log "Tracking resource: $1"
}

# Function to check if command output contains actual errors
check_for_errors() {
    local output="$1"
    local command_name="$2"

    # Check for specific AWS CLI error patterns, not just any occurrence of
 "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameterException\|
AccessDenied\|ValidationException\|ResourceNotFoundException"; then
        log "ERROR in $command_name: $output"
```

```
        return 1
    fi
    return 0
}

# Function to get AWS account ID
get_account_id() {
    ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
    log "Using AWS Account ID: $ACCOUNT_ID"
}

# Function to wait for resources to be ready
wait_for_resource() {
    local resource_type="$1"
    local resource_id="$2"

    case "$resource_type" in
        "cluster")
            log "Waiting for cluster $resource_id to be active..."
            local attempt=1
            local max_attempts=30
            while [ $attempt -le $max_attempts ]; do
                local status=$(aws ecs describe-clusters --clusters
 "$resource_id" --query 'clusters[0].status' --output text)
                if [ "$status" = "ACTIVE" ]; then
                    log "Cluster is now active"
                    return 0
                fi
                log "Cluster status: $status (attempt $attempt/$max_attempts)"
                sleep 10
                ((attempt++))
            done
            log "ERROR: Cluster did not become active within expected time"
            return 1
            ;;
        "service")
            log "Waiting for service $resource_id to be stable..."
            aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id"
            ;;
        "nat-gateway")
            log "Waiting for NAT Gateway $resource_id to be available..."
            aws ec2 wait nat-gateway-available --nat-gateway-ids "$resource_id"
            ;;
```

```
    esac
}

# Function to use default VPC infrastructure
setup_default_vpc_infrastructure() {
    log "Using default VPC infrastructure..."

    # Get default VPC
    VPC_ID=$(aws ec2 describe-vpcs --filters "Name=isDefault,Values=true" --query
 'Vpcs[0].VpcId' --output text)
    if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
        log "ERROR: No default VPC found. Please create a default VPC first."
        exit 1
    fi
    log "Using default VPC: $VPC_ID"

    # Get default subnets
    SUBNETS=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 "Name=default-for-az,Values=true" --query 'Subnets[].SubnetId' --output text)
    SUBNET_ARRAY=($SUBNETS)

    if [ ${#SUBNET_ARRAY[@]} -lt 2 ]; then
        log "ERROR: Need at least 2 subnets for ECS Service Connect. Found:
 ${#SUBNET_ARRAY[@]}"
        exit 1
    fi

    PUBLIC_SUBNET1=${SUBNET_ARRAY[0]}
    PUBLIC_SUBNET2=${SUBNET_ARRAY[1]}

    log "Using subnets: $PUBLIC_SUBNET1, $PUBLIC_SUBNET2"

    # Create security group for ECS tasks
    SG_OUTPUT=$(aws ec2 create-security-group \
        --group-name "${ENV_PREFIX}-ecs-sg-${RANDOM_SUFFIX}" \
        --description "Security group for ECS Service Connect tutorial" \
        --vpc-id "$VPC_ID" \
        --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-ecs-service-
connect}]' 2>&1)
    check_for_errors "$SG_OUTPUT" "create-security-group"
    SECURITY_GROUP_ID=$(echo "$SG_OUTPUT" | grep -o '"GroupId": "[^"]*"' | cut -
d'"' -f4)
    track_resource "SG:$SECURITY_GROUP_ID"
```

```
    log "Created security group: $SECURITY_GROUP_ID"

    # Add inbound rules to security group
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 443 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    log "Default VPC infrastructure setup completed"
}

# Function to create CloudWatch log groups
create_log_groups() {
    log "Creating CloudWatch log groups..."

    # Create log group for nginx container
    aws logs create-log-group --log-group-name "/ecs/service-connect-nginx"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-nginx created"
            track_resource "LOG_GROUP:/ecs/service-connect-nginx"
        else
            log "Log group /ecs/service-connect-nginx already exists"
        fi
    }

    # Create log group for service connect proxy
    aws logs create-log-group --log-group-name "/ecs/service-connect-proxy"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-proxy created"
            track_resource "LOG_GROUP:/ecs/service-connect-proxy"
        else
            log "Log group /ecs/service-connect-proxy already exists"
        fi
```

```
    }
}

# Function to create ECS cluster with Service Connect
create_ecs_cluster() {
    log "Creating ECS cluster with Service Connect..."

    CLUSTER_OUTPUT=$(aws ecs create-cluster \
        --cluster-name "$CLUSTER_NAME" \
        --service-connect-defaults namespace="$NAMESPACE_NAME" \
        --tags key=Environment,value=tutorial key=project,value=doc-smith
 key=tutorial,value=amazon-ecs-service-connect 2>&1)
    check_for_errors "$CLUSTER_OUTPUT" "create-cluster"

    track_resource "CLUSTER:$CLUSTER_NAME"
    log "Created ECS cluster: $CLUSTER_NAME"

    wait_for_resource "cluster" "$CLUSTER_NAME"

    # Track the Service Connect namespace that gets created
    # Wait a moment for the namespace to be created
    sleep 5
    NAMESPACE_ID=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_ID" && "$NAMESPACE_ID" != "None" ]]; then
        track_resource "NAMESPACE:$NAMESPACE_ID"
        log "Service Connect namespace created: $NAMESPACE_ID"
    fi
}

# Function to create IAM roles
create_iam_roles() {
    log "Creating IAM roles..."

    # Check if ecsTaskExecutionRole exists
    if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
        log "IAM role ecsTaskExecutionRole exists"
    else
        log "Creating ecsTaskExecutionRole..."
        aws iam create-role \
            --role-name ecsTaskExecutionRole \
```

```
            --assume-role-policy-document '{
                "Version":"2012-10-17",
                "Statement": [{
                    "Effect": "Allow",
                    "Principal": {"Service": "ecs-tasks.amazonaws.com"},
                    "Action": "sts:AssumeRole"
                }]
            }' >/dev/null 2>&1
        aws iam attach-role-policy \
            --role-name ecsTaskExecutionRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy >/dev/null 2>&1
        track_resource "ROLE:ecsTaskExecutionRole"
        aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-service-connect
        log "Created ecsTaskExecutionRole"
        sleep 10
    fi

    # Check if ecsTaskRole exists, create if not
    if aws iam get-role --role-name ecsTaskRole >/dev/null 2>&1; then
        log "IAM role ecsTaskRole exists"
    else
        log "IAM role ecsTaskRole does not exist, will create it"

        # Create trust policy for ECS tasks
        cat > /tmp/ecs-task-trust-policy.json << EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

        aws iam create-role \
            --role-name ecsTaskRole \
```

```
            --assume-role-policy-document file:///tmp/ecs-task-trust-policy.json
 >/dev/null

        track_resource "IAM_ROLE:ecsTaskRole"
        aws iam tag-role --role-name ecsTaskRole --tags Key=project,Value=doc-
smith Key=tutorial,Value=amazon-ecs-service-connect
        log "Created ecsTaskRole"

        # Wait for role to be available
        sleep 10
    fi
}

# Function to create task definition
create_task_definition() {
    log "Creating task definition..."

    # Create task definition JSON
    cat > /tmp/task-definition.json << EOF
{
    "family": "service-connect-nginx",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole",
    "taskRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskRole",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "portMappings": [
                {
                    "containerPort": 80,
                    "protocol": "tcp",
                    "name": "nginx-port"
                }
            ],
            "essential": true,
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/service-connect-nginx",
                    "awslogs-region": "${REGION}",
```

```
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

    TASK_DEF_OUTPUT=$(aws ecs register-task-definition --cli-input-json file:///
tmp/task-definition.json 2>&1)
    check_for_errors "$TASK_DEF_OUTPUT" "register-task-definition"

    TASK_DEF_ARN=$(echo "$TASK_DEF_OUTPUT" | grep -o '"taskDefinitionArn":
 "[^"]*"' | cut -d'"' -f4)
    track_resource "TASK_DEF:service-connect-nginx"
    log "Created task definition: $TASK_DEF_ARN"

    # Clean up temporary file
    rm -f /tmp/task-definition.json
}

# Function to create ECS service with Service Connect
create_ecs_service() {
    log "Creating ECS service with Service Connect..."

    # Create service definition JSON
    cat > /tmp/service-definition.json << EOF
{
    "serviceName": "service-connect-nginx-service",
    "cluster": "${CLUSTER_NAME}",
    "taskDefinition": "service-connect-nginx",
    "desiredCount": 1,
    "launchType": "FARGATE",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": ["${PUBLIC_SUBNET1}", "${PUBLIC_SUBNET2}"],
            "securityGroups": ["${SECURITY_GROUP_ID}"],
            "assignPublicIp": "ENABLED"
        }
    },
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "${NAMESPACE_NAME}",
        "services": [
```

```
            {
                "portName": "nginx-port",
                "discoveryName": "nginx",
                "clientAliases": [
                    {
                        "port": 80,
                        "dnsName": "nginx"
                    }
                ]
            }
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-proxy",
                "awslogs-region": "${REGION}",
                "awslogs-stream-prefix": "ecs-service-connect"
            }
        }
    },
    "tags": [
        {
            "key": "Environment",
            "value": "tutorial"
        },
        {
            "key": "project",
            "value": "doc-smith"
        },
        {
            "key": "tutorial",
            "value": "amazon-ecs-service-connect"
        }
    ]
}
EOF

    SERVICE_OUTPUT=$(aws ecs create-service --cli-input-json file:///tmp/service-
definition.json 2>&1)
    check_for_errors "$SERVICE_OUTPUT" "create-service"

    track_resource "SERVICE:service-connect-nginx-service"
    log "Created ECS service: service-connect-nginx-service"
```

```
    wait_for_resource "service" "service-connect-nginx-service"

    # Clean up temporary file
    rm -f /tmp/service-definition.json
}

# Function to verify deployment
verify_deployment() {
    log "Verifying deployment..."

    # Check service status
    SERVICE_STATUS=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].status' --output text)
    log "Service status: $SERVICE_STATUS"

    # Check running tasks
    RUNNING_COUNT=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].runningCount' --output text)
    log "Running tasks: $RUNNING_COUNT"

    # Get task ARN
    TASK_ARN=$(aws ecs list-tasks \
        --cluster "$CLUSTER_NAME" \
        --service-name "service-connect-nginx-service" \
        --query 'taskArns[0]' --output text)

    if [[ "$TASK_ARN" != "None" && -n "$TASK_ARN" ]]; then
        log "Task ARN: $TASK_ARN"

        # Try to get task IP address
        TASK_IP=$(aws ecs describe-tasks \
            --cluster "$CLUSTER_NAME" \
            --tasks "$TASK_ARN" \
            --query 'tasks[0].attachments[0].details[?
name==`privateIPv4Address`].value' \
            --output text 2>/dev/null || echo "")

        if [[ -n "$TASK_IP" && "$TASK_IP" != "None" ]]; then
            log "Task IP address: $TASK_IP"
        else
```

```
            log "Could not retrieve task IP address"
        fi
    fi

    # Check Service Connect namespace
    NAMESPACE_STATUS=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_STATUS" && "$NAMESPACE_STATUS" != "None" ]]; then
        log "Service Connect namespace '$NAMESPACE_NAME' is active"
    else
        log "Service Connect namespace '$NAMESPACE_NAME' not found or not active"
    fi

    # Display Service Connect configuration
    log "Service Connect configuration:"
    aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].serviceConnectConfiguration' 2>/dev/null || true
}

# Function to display created resources
display_resources() {
    echo ""
    echo "============================================="
    echo "CREATED RESOURCES"
    echo "============================================="
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "- $resource"
    done
    echo "============================================="
    echo ""
}

# Function to cleanup resources
cleanup_resources() {
    log "Starting cleanup process..."

    # Delete resources in reverse order of creation
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[i]}"
```

```
        resource_type=$(echo "$resource" | cut -d':' -f1)
        resource_id=$(echo "$resource" | cut -d':' -f2)

        log "Cleaning up $resource_type: $resource_id"

        case "$resource_type" in
            "SERVICE")
                aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --desired-count 0 2>&1 | grep -qi "error" && log "Warning: Failed
 to scale down service $resource_id"
                aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id" 2>/dev/null || true
                aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --force 2>&1 | grep -qi "error" && log "Warning: Failed to delete
 service $resource_id"
                ;;
            "TASK_DEF")
                TASK_DEF_ARNS=$(aws ecs list-task-definitions --family-prefix
 "$resource_id" --query 'taskDefinitionArns' --output text 2>/dev/null)
                for arn in $TASK_DEF_ARNS; do
                    aws ecs deregister-task-definition --task-definition "$arn"
 >/dev/null 2>&1 || true
                done
                ;;
            "ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "IAM_ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "CLUSTER")
                aws ecs delete-cluster --cluster "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete cluster $resource_id"
                ;;
            "SG")
                for attempt in 1 2 3 4 5; do
```

```
                    if aws ec2 delete-security-group --group-id "$resource_id"
 2>/dev/null; then
                        break
                    fi
                    log "Security group $resource_id still has dependencies,
 retrying in 30s ($attempt/5)..."
                    sleep 30
                done
                ;;
            "LOG_GROUP")
                aws logs delete-log-group --log-group-name "$resource_id" 2>&1 |
 grep -qi "error" && log "Warning: Failed to delete log group $resource_id"
                ;;
            "NAMESPACE")
                # First, delete any services in the namespace
                NAMESPACE_SERVICES=$(aws servicediscovery list-services \
                    --filters Name=NAMESPACE_ID,Values="$resource_id" \
                    --query 'Services[].Id' --output text 2>/dev/null || echo "")

                if [[ -n "$NAMESPACE_SERVICES" && "$NAMESPACE_SERVICES" !=
 "None" ]]; then
                    for service_id in $NAMESPACE_SERVICES; do
                        aws servicediscovery delete-service --id "$service_id" >/
dev/null 2>&1 || true
                        sleep 2
                    done
                fi

                # Then delete the namespace
                aws servicediscovery delete-namespace --id "$resource_id" >/dev/
null 2>&1 || true
                ;;
        esac

        sleep 2  # Brief pause between deletions
    done

    # Clean up temporary files
    rm -f /tmp/ecs-task-trust-policy.json
    rm -f /tmp/task-definition.json
    rm -f /tmp/service-definition.json

    log "Cleanup completed"
}
```

```
# Main execution
main() {
    log "Starting $SCRIPT_NAME v4 (Default VPC)"
    log "Region: $REGION"
    log "Log file: $LOG_FILE"

    # Get AWS account ID
    get_account_id

    # Setup infrastructure using default VPC
    setup_default_vpc_infrastructure

    # Create CloudWatch log groups
    create_log_groups

    # Create ECS cluster
    create_ecs_cluster

    # Create IAM roles
    create_iam_roles

    # Create task definition
    create_task_definition

    # Create ECS service
    create_ecs_service

    # Verify deployment
    verify_deployment

    log "Tutorial completed successfully!"

    # Display created resources
    display_resources

    # Ask user if they want to clean up
    echo ""
    echo "============================================"
    echo "CLEANUP CONFIRMATION"
    echo "============================================"
    echo "Do you want to clean up all created resources? (y/n): "
    CLEANUP_CHOICE="y"
```

```
    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        cleanup_resources
        log "All resources have been cleaned up"
    else
        log "Resources left intact. You can clean them up later by running the
 cleanup function."
        echo ""
        echo "To clean up resources later, you can use the AWS CLI commands or
 the AWS Management Console."
        echo "Remember to delete resources in the correct order to avoid
 dependency issues."
    fi
}

# Make script executable and run
chmod +x "$0"
main "$@"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateLogGroup
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteLogGroup
  - DeleteNamespace
  - DeleteRole
  - DeleteSecurityGroup
  - DeleteService (AWS Cloud Map)
  - DeleteService (Amazon ECS)
  - DeregisterTaskDefinition

- DescribeClusters
- DescribeServices
- DescribeSubnets
- DescribeTasks
- DescribeVpcs
- DetachRolePolicy
- GetCallerIdentity
- GetRole
- ListNamespaces
- ListServices
- ListTaskDefinitions
- ListTasks
- RegisterTaskDefinition
- UpdateService
- Attendre (Amazon EC2)
- Attendre (Amazon ECS)

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Construisez une URL avec AWS STS pour les utilisateurs fédérés utilisant un AWS Kit SDK

L'exemple de code suivant illustre comment :

- Créez un rôle IAM qui accorde un accès en lecture seule aux ressources Amazon S3 du compte actuel.
- Obtenez un jeton de sécurité auprès du point de terminaison AWS de la fédération.
- Créez une URL qui puisse être utilisée pour accéder à la console avec des informations d'identification fédérées.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez un rôle qui accorde un accès en lecture seule aux ressources Amazon S3 du compte actuel.

```
def setup(iam_resource):
    """
    Creates a role that can be assumed by the current user.
    Attaches a policy that allows only Amazon S3 read-only access.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 instance
                         that has the permission to create a role.
    :return: The newly created role.
    """
    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": iam_resource.CurrentUser().arn},
                        "Action": "sts:AssumeRole",
                    }
                ],
            }
        ),
    )
    role.attach_policy(PolicyArn="arn:aws:iam::aws:policy/
AmazonS3ReadOnlyAccess")
    print(f"Created role {role.name}.")
```

```
    print("Give AWS time to propagate these new resources and connections.",
end="")
    progress_bar(10)

    return role
```

Obtenez un jeton de sécurité auprès du point de terminaison de la AWS fédération et créez une URL qui peut être utilisée pour accéder à la console avec des informations d'identification fédérées.

```
def construct_federated_url(assume_role_arn, session_name, issuer, sts_client):
    """
    Constructs a URL that gives federated users direct access to the AWS
Management
    Console.

    1. Acquires temporary credentials from AWS Security Token Service (AWS STS)
that
       can be used to assume a role with limited permissions.
    2. Uses the temporary credentials to request a sign-in token from the
       AWS federation endpoint.
    3. Builds a URL that can be used in a browser to navigate to the AWS
federation
       endpoint, includes the sign-in token for authentication, and redirects to
       the AWS Management Console with permissions defined by the role that was
       specified in step 1.

    :param assume_role_arn: The role that specifies the permissions that are
granted.
                            The current user must have permission to assume the
role.
    :param session_name: The name for the STS session.
    :param issuer: The organization that issues the URL.
    :param sts_client: A Boto3 STS instance that can assume the role.
    :return: The federated URL.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn, RoleSessionName=session_name
    )
    temp_credentials = response["Credentials"]
```

```
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    session_data = {
        "sessionId": temp_credentials["AccessKeyId"],
        "sessionKey": temp_credentials["SecretAccessKey"],
        "sessionToken": temp_credentials["SessionToken"],
    }
    aws_federated_signin_endpoint = "https://signin.aws.amazon.com/federation"

    # Make a request to the AWS federation endpoint to get a sign-in token.
    # The requests.get function URL-encodes the parameters and builds the query
string
    # before making the request.
    response = requests.get(
        aws_federated_signin_endpoint,
        params={
            "Action": "getSigninToken",
            "SessionDuration": str(datetime.timedelta(hours=12).seconds),
            "Session": json.dumps(session_data),
        },
    )
    signin_token = json.loads(response.text)
    print(f"Got a sign-in token from the AWS sign-in federation endpoint.")

    # Make a federated URL that can be used to sign into the AWS Management
Console.
    query_string = urllib.parse.urlencode(
        {
            "Action": "login",
            "Issuer": issuer,
            "Destination": "https://console.aws.amazon.com/",
            "SigninToken": signin_token["SigninToken"],
        }
    )
    federated_url = f"{aws_federated_signin_endpoint}?{query_string}"
    return federated_url
```

Détruisez les ressources créées pour la démo.

```
def teardown(role):
```

```
    """
    Removes all resources created during setup.

    :param role: The demo role.
    """
    for attached in role.attached_policies.all():
        role.detach_policy(PolicyArn=attached.arn)
        print(f"Detached {attached.policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
```

Exécutez ce scénario à l'aide des fonctions définies précédemment.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the AWS Security Token Service federated URL demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    role = setup(iam_resource)
    sts_client = boto3.client("sts")
    try:
        federated_url = construct_federated_url(
            role.arn, "AssumeRoleDemoSession", "example.org", sts_client
        )
        print(
            "Constructed a federated URL that can be used to connect to the "
            "AWS Management Console with role-defined permissions:"
        )
        print("-" * 88)
        print(federated_url)
        print("-" * 88)
        _ = input(
            "Copy and paste the above URL into a browser to open the AWS "
            "Management Console with limited permissions. When done, press "
            "Enter to clean up and complete this demo."
        )
    finally:
        teardown(role)
        print("Thanks for watching!")
```

- Pour plus de détails sur l'API, consultez AssumeRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Création d'une tâche de conteneur pour le type de lancement sans serveur

L'exemple de code suivant illustre comment :

- Créer le cluster
- Créer une définition de tâche
- Créez le service
- Nettoyage

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Amazon ECS Fargate Tutorial Script - Version 5
# This script creates an ECS cluster, task definition, and service using Fargate
 launch type
# Fixed version with proper resource dependency handling during cleanup
# Security improvements applied

set -e  # Exit on any error
```

```
# Initialize logging
LOG_FILE="ecs-fargate-tutorial-v5.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon ECS Fargate tutorial at $(date)"
echo "Log file: $LOG_FILE"

# Generate random identifier for unique resource names
RANDOM_ID=$(openssl rand -hex 6)
CLUSTER_NAME="fargate-cluster-${RANDOM_ID}"
SERVICE_NAME="fargate-service-${RANDOM_ID}"
TASK_FAMILY="sample-fargate-${RANDOM_ID}"
SECURITY_GROUP_NAME="ecs-fargate-sg-${RANDOM_ID}"

# Array to track created resources for cleanup
CREATED_RESOURCES=()

# Function to log and execute commands with input validation
execute_command() {
    local cmd="$1"
    local description="$2"

    # Validate that cmd is not empty
    if [[ -z "$cmd" ]]; then
        echo "ERROR: Command is empty"
        return 1
    fi

    echo ""
    echo "================================================"
    echo "EXECUTING: $description"
    echo "COMMAND: $cmd"
    echo "================================================"

    local output
    local exit_code
    set +e  # Temporarily disable exit on error
    output=$(eval "$cmd" 2>&1)
    exit_code=$?
    set -e  # Re-enable exit on error

    if [[ $exit_code -eq 0 ]]; then
        echo "SUCCESS: $description"
```

```
        echo "OUTPUT: $output"
        return 0
    else
        echo "FAILED: $description"
        echo "EXIT CODE: $exit_code"
        echo "OUTPUT: $output"
        return 1
    fi
}

# Function to check for actual AWS API errors in command output
check_for_aws_errors() {
    local output="$1"
    local description="$2"

    # Look for specific AWS error patterns, not just the word "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameter\|
AccessDenied\|ResourceNotFound\|ValidationException"; then
        echo "AWS API ERROR detected in output for: $description"
        echo "Output: $output"
        return 1
    fi
    return 0
}

# Function to safely extract JSON values
safe_json_extract() {
    local json="$1"
    local key="$2"
    local value

    value=$(echo "$json" | grep -o "\"$key\": \"[^\"]*\"" | cut -d'"' -f4 2>/dev/
null || echo "")
    echo "$value"
}

# Function to wait for network interfaces to be cleaned up
wait_for_network_interfaces_cleanup() {
    local security_group_id="$1"
    local max_attempts=30
    local attempt=1

    # Validate security group ID format
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
```

```
        echo "ERROR: Invalid security group ID format: $security_group_id"
        return 1
    fi

    echo "Waiting for network interfaces to be cleaned up..."

    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Checking for dependent network
 interfaces..."

        # Check if there are any network interfaces still using this security
 group
        local eni_count
        eni_count=$(aws ec2 describe-network-interfaces \
            --filters "Name=group-id,Values=$security_group_id" \
            --query "length(NetworkInterfaces)" \
            --output text 2>/dev/null || echo "0")

        if [[ "$eni_count" == "0" ]]; then
            echo "No network interfaces found using security group
 $security_group_id"
            return 0
        else
            echo "Found $eni_count network interface(s) still using security
 group $security_group_id"
            echo "Waiting 10 seconds before next check..."
            sleep 10
            ((attempt++))
        fi
    done

    echo "WARNING: Network interfaces may still be attached after $max_attempts
 attempts"
    echo "This is normal and the security group deletion will be retried"
    return 1
}

# Function to retry security group deletion with exponential backoff
retry_security_group_deletion() {
    local security_group_id="$1"
    local max_attempts=10
    local attempt=1
    local wait_time=5
```

```
    # Validate security group ID format
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
        echo "ERROR: Invalid security group ID format: $security_group_id"
        return 1
    fi

    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Trying to delete security group
 $security_group_id"

        if execute_command "aws ec2 delete-security-group --group-id
 '$security_group_id'" "Delete security group (attempt $attempt)"; then
            echo "Successfully deleted security group $security_group_id"
            return 0
        else
            if [[ $attempt -eq $max_attempts ]]; then
                echo "FAILED: Could not delete security group $security_group_id
 after $max_attempts attempts"
                echo "This may be due to network interfaces that are still being
 cleaned up by AWS"
                echo "You can manually delete it later using: aws ec2 delete-
security-group --group-id $security_group_id"
                return 1
            else
                echo "Waiting $wait_time seconds before retry..."
                sleep $wait_time
                wait_time=$((wait_time * 2))  # Exponential backoff
                ((attempt++))
            fi
        fi
    done
}

# Function to cleanup resources with proper dependency handling
cleanup_resources() {
    echo ""
    echo "========================================="
    echo "CLEANUP PROCESS"
    echo "========================================="
    echo "The following resources were created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
    echo ""
```

```
   echo "Auto-confirming cleanup of all created resources..."

   CLEANUP_CHOICE="y"

   if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
       echo "Starting cleanup process..."

       # Step 1: Scale service to 0 tasks first, then delete service
       if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Service: $SERVICE_NAME " ]];
then
           echo ""
           echo "Step 1: Scaling service to 0 tasks..."
           if execute_command "aws ecs update-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME' --desired-count 0" "Scale service to 0 tasks"; then
               echo "Waiting for service to stabilize after scaling to 0..."
               execute_command "aws ecs wait services-stable --cluster
'$CLUSTER_NAME' --services '$SERVICE_NAME'" "Wait for service to stabilize"

               echo "Deleting service..."
               execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME'" "Delete ECS service"
           else
               echo "WARNING: Failed to scale service. Attempting to delete
anyway..."
               execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME' --force" "Force delete ECS service"
           fi
       fi

       # Step 2: Wait a bit for tasks to fully terminate
       echo ""
       echo "Step 2: Waiting for tasks to fully terminate..."
       sleep 15

       # Step 3: Delete cluster
       if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Cluster: $CLUSTER_NAME " ]];
then
           echo ""
           echo "Step 3: Deleting cluster..."
           execute_command "aws ecs delete-cluster --cluster '$CLUSTER_NAME'"
"Delete ECS cluster"
       fi
```

```
        # Step 4: Wait for network interfaces to be cleaned up, then delete
 security group
        if [[ -n "$SECURITY_GROUP_ID" && "$SECURITY_GROUP_ID" != "None" ]]; then
            echo ""
            echo "Step 4: Cleaning up security group..."

            # First, wait for network interfaces to be cleaned up
            wait_for_network_interfaces_cleanup "$SECURITY_GROUP_ID"

            # Then retry security group deletion with backoff
            retry_security_group_deletion "$SECURITY_GROUP_ID"
        fi

        # Step 5: Clean up task definition (deregister all revisions)
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
 " ]]; then
            echo ""
            echo "Step 5: Deregistering task definition revisions..."

            # Get all revisions of the task definition
            local revisions
            revisions=$(aws ecs list-task-definitions --family-prefix
 "$TASK_FAMILY" --query "taskDefinitionArns" --output text 2>/dev/null || echo
 "")

            if [[ -n "$revisions" && "$revisions" != "None" ]]; then
                for revision_arn in $revisions; do
                    echo "Deregistering task definition: $revision_arn"
                    execute_command "aws ecs deregister-task-definition --task-
definition '$revision_arn'" "Deregister task definition $revision_arn" || true
                done
            else
                echo "No task definition revisions found to deregister"
            fi
        fi

        echo ""
        echo "============================================"
        echo "CLEANUP COMPLETED"
        echo "============================================"
        echo "All resources have been cleaned up successfully!"

    else
        echo "Cleanup skipped. Resources remain active."
```

```
        echo ""
        echo "To clean up manually later, use the following commands in order:"
        echo "1. Scale service to 0: aws ecs update-service --cluster
 $CLUSTER_NAME --service $SERVICE_NAME --desired-count 0"
        echo "2. Wait for stability: aws ecs wait services-stable --cluster
 $CLUSTER_NAME --services $SERVICE_NAME"
        echo "3. Delete service: aws ecs delete-service --cluster $CLUSTER_NAME
 --service $SERVICE_NAME"
        echo "4. Delete cluster: aws ecs delete-cluster --cluster $CLUSTER_NAME"
        echo "5. Wait 2-3 minutes, then delete security group: aws ec2 delete-
security-group --group-id $SECURITY_GROUP_ID"
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
 " ]]; then
            echo "6. Deregister task definitions: aws ecs list-task-definitions
 --family-prefix $TASK_FAMILY"
            echo "   Then for each ARN: aws ecs deregister-task-definition --
task-definition <ARN>"
        fi
    fi
}

# Trap to handle script interruption
trap cleanup_resources EXIT

echo "Using random identifier: $RANDOM_ID"
echo "Cluster name: $CLUSTER_NAME"
echo "Service name: $SERVICE_NAME"
echo "Task family: $TASK_FAMILY"

# Step 1: Ensure ECS task execution role exists
echo ""
echo "==========================================="
echo "STEP 1: VERIFY ECS TASK EXECUTION ROLE"
echo "==========================================="

ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ ! "$ACCOUNT_ID" =~ ^[0-9]{12}$ ]]; then
    echo "ERROR: Invalid AWS Account ID retrieved: $ACCOUNT_ID"
    exit 1
fi

EXECUTION_ROLE_ARN="arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole"

# Check if role exists
```

```
if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
    echo "ECS task execution role already exists"
else
    echo "Creating ECS task execution role..."

    # Create trust policy with strict validation
    cat > trust-policy.json << 'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

    # Validate JSON before using
    if ! jq empty trust-policy.json 2>/dev/null; then
        echo "ERROR: Invalid JSON in trust policy"
        rm -f trust-policy.json
        exit 1
    fi

    execute_command "aws iam create-role --role-name ecsTaskExecutionRole --
assume-role-policy-document file://trust-policy.json" "Create ECS task execution
 role"

    aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-fargate-linux

    execute_command "aws iam attach-role-policy --role-name
 ecsTaskExecutionRole --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy" "Attach ECS task execution policy"

    # Clean up temporary file securely
    shred -vfz -n 3 trust-policy.json 2>/dev/null || rm -f trust-policy.json

    CREATED_RESOURCES+=("IAM Role: ecsTaskExecutionRole")
fi
```

```
# Step 2: Create ECS cluster
echo ""
echo "============================================"
echo "STEP 2: CREATE ECS CLUSTER"
echo "============================================"

CLUSTER_OUTPUT=$(execute_command "aws ecs create-cluster --cluster-name
 '$CLUSTER_NAME' --tags key=project,value=doc-smith key=tutorial,value=amazon-
ecs-fargate-linux" "Create ECS cluster")
check_for_aws_errors "$CLUSTER_OUTPUT" "Create ECS cluster"

CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")

# Step 3: Create task definition
echo ""
echo "============================================"
echo "STEP 3: CREATE TASK DEFINITION"
echo "============================================"

# Create task definition JSON with validated inputs
cat > task-definition.json << EOF
{
    "family": "$TASK_FAMILY",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "$EXECUTION_ROLE_ARN",
    "containerDefinitions": [
        {
            "name": "fargate-app",
            "image": "public.ecr.aws/docker/library/httpd:2.4-alpine",
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": ["sh", "-c"],
            "command": [
```

```
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container
 in Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html
 && httpd-foreground\""
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/fargate-sample",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

# Validate JSON before using
if ! jq empty task-definition.json 2>/dev/null; then
    echo "ERROR: Invalid JSON in task definition"
    rm -f task-definition.json
    exit 1
fi

TASK_DEF_OUTPUT=$(execute_command "aws ecs register-task-definition --cli-input-
json file://task-definition.json" "Register task definition")
check_for_aws_errors "$TASK_DEF_OUTPUT" "Register task definition"

# Clean up temporary file securely
shred -vfz -n 3 task-definition.json 2>/dev/null || rm -f task-definition.json

CREATED_RESOURCES+=("Task Definition: $TASK_FAMILY")

# Step 4: Set up networking
echo ""
echo "==========================================="
echo "STEP 4: SET UP NETWORKING"
echo "==========================================="

# Get default VPC ID
```

```
VPC_ID=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true" --query
 "Vpcs[0].VpcId" --output text)
if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
    echo "ERROR: No default VPC found. Please create a default VPC or specify a
 custom VPC."
    exit 1
fi

# Validate VPC ID format
if [[ ! "$VPC_ID" =~ ^vpc-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid VPC ID format: $VPC_ID"
    exit 1
fi

echo "Using default VPC: $VPC_ID"

# Create security group with restricted access
# Note: This allows HTTP access from anywhere for demo purposes
# In production, restrict source to specific IP ranges or security groups
SECURITY_GROUP_OUTPUT=$(execute_command "aws ec2 create-security-group --
group-name '$SECURITY_GROUP_NAME' --description 'Security group for ECS
 Fargate tutorial - HTTP access' --vpc-id '$VPC_ID' --tag-specifications
 'ResourceType=security-group,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-ecs-fargate-linux}]'" "Create security group")
check_for_aws_errors "$SECURITY_GROUP_OUTPUT" "Create security group"

SECURITY_GROUP_ID=$(echo "$SECURITY_GROUP_OUTPUT" | grep -o '"GroupId": "[^"]*"'
 | head -1 | cut -d'"' -f4)
if [[ -z "$SECURITY_GROUP_ID" || "$SECURITY_GROUP_ID" == "None" ]]; then
    SECURITY_GROUP_ID=$(aws ec2 describe-security-groups --group-names
 "$SECURITY_GROUP_NAME" --filters "Name=vpc-id,Values=$VPC_ID" --query
 "SecurityGroups[0].GroupId" --output text)
fi

# Validate security group ID format
if [[ ! "$SECURITY_GROUP_ID" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid security group ID format: $SECURITY_GROUP_ID"
    exit 1
fi

echo "Created security group: $SECURITY_GROUP_ID"
CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")

# Add HTTP inbound rule
```

```
# WARNING: This allows HTTP access from anywhere (0.0.0.0/0)
# In production environments, restrict this to specific IP ranges
execute_command "aws ec2 authorize-security-group-ingress --group-id
 '$SECURITY_GROUP_ID' --protocol tcp --port 80 --cidr 0.0.0.0/0" "Add HTTP
 inbound rule to security group"

# Get subnet IDs from default VPC
echo "Getting subnet IDs from default VPC..."
SUBNET_IDS_RAW=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 --query "Subnets[*].SubnetId" --output text)
if [[ -z "$SUBNET_IDS_RAW" ]]; then
    echo "ERROR: No subnets found in default VPC"
    exit 1
fi

# Convert to proper comma-separated format, handling both spaces and tabs
SUBNET_IDS_COMMA=$(echo "$SUBNET_IDS_RAW" | tr -s '[:space:]' ',' | sed 's/,$//')
echo "Raw subnet IDs: $SUBNET_IDS_RAW"
echo "Formatted subnet IDs: $SUBNET_IDS_COMMA"

# Validate subnet IDs format
if [[ ! "$SUBNET_IDS_COMMA" =~ ^subnet-[a-z0-9]+(,subnet-[a-z0-9]+)*$ ]]; then
    echo "ERROR: Invalid subnet ID format: $SUBNET_IDS_COMMA"
    exit 1
fi

# Step 5: Create ECS service
echo ""
echo "==========================================="
echo "STEP 5: CREATE ECS SERVICE"
echo "==========================================="

# Create the service with proper JSON formatting for network configuration
SERVICE_CMD="aws ecs create-service --cluster '$CLUSTER_NAME' --service-name
 '$SERVICE_NAME' --task-definition '$TASK_FAMILY' --desired-count 1 --launch-
type FARGATE --network-configuration '{\"awsvpcConfiguration\":{\"subnets
\":[\"$(echo "$SUBNET_IDS_COMMA" | sed 's/,/","/g')\"],\"securityGroups
\":[\"$SECURITY_GROUP_ID\"],\"assignPublicIp\":\"ENABLED\"}}' --tags
 key=project,value=doc-smith key=tutorial,value=amazon-ecs-fargate-linux"

echo "Service creation command: $SERVICE_CMD"

SERVICE_OUTPUT=$(execute_command "$SERVICE_CMD" "Create ECS service")
check_for_aws_errors "$SERVICE_OUTPUT" "Create ECS service"
```

```
CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")

# Step 6: Wait for service to stabilize and get public IP
echo ""
echo "============================================="
echo "STEP 6: WAIT FOR SERVICE AND GET PUBLIC IP"
echo "============================================="

echo "Waiting for service to stabilize (this may take a few minutes)..."
execute_command "aws ecs wait services-stable --cluster '$CLUSTER_NAME' --
services '$SERVICE_NAME'" "Wait for service to stabilize"

# Get task ARN
TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query "taskArns[0]" --output text)
if [[ "$TASK_ARN" == "None" || -z "$TASK_ARN" ]]; then
    echo "ERROR: No running tasks found for service"
    exit 1
fi

echo "Task ARN: $TASK_ARN"

# Get network interface ID
ENI_ID=$(aws ecs describe-tasks --cluster "$CLUSTER_NAME" --tasks "$TASK_ARN"
 --query "tasks[0].attachments[0].details[?name=='networkInterfaceId'].value" --
output text)
if [[ "$ENI_ID" == "None" || -z "$ENI_ID" ]]; then
    echo "ERROR: Could not retrieve network interface ID"
    exit 1
fi

# Validate ENI ID format
if [[ ! "$ENI_ID" =~ ^eni-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid network interface ID format: $ENI_ID"
    exit 1
fi

echo "Network Interface ID: $ENI_ID"

# Get public IP
PUBLIC_IP=$(aws ec2 describe-network-interfaces --network-interface-ids "$ENI_ID"
 --query "NetworkInterfaces[0].Association.PublicIp" --output text)
if [[ "$PUBLIC_IP" == "None" || -z "$PUBLIC_IP" ]]; then
```

```
    echo "WARNING: No public IP assigned to the task"
    echo "The task may be in a private subnet or public IP assignment failed"
else
    # Validate IP format
    if [[ ! "$PUBLIC_IP" =~ ^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$ ]];
 then
        echo "ERROR: Invalid IP address format: $PUBLIC_IP"
        PUBLIC_IP=""
    else
        echo ""
        echo "=========================================="
        echo "SUCCESS! APPLICATION IS RUNNING"
        echo "=========================================="
        echo "Your application is available at: http://$PUBLIC_IP"
        echo "You can test it by opening this URL in your browser"
        echo ""
    fi
fi

# Display service information
echo ""
echo "=========================================="
echo "SERVICE INFORMATION"
echo "=========================================="
execute_command "aws ecs describe-services --cluster '$CLUSTER_NAME' --services
 '$SERVICE_NAME'" "Get service details"

echo ""
echo "=========================================="
echo "TUTORIAL COMPLETED SUCCESSFULLY"
echo "=========================================="
echo "Resources created:"
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "  - $resource"
done

if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
    echo ""
    echo "Application URL: http://$PUBLIC_IP"
fi

echo ""
echo "Script completed at $(date)"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteSecurityGroup
  - DeleteService
  - DeregisterTaskDefinition
  - DescribeNetworkInterfaces
  - DescribeSecurityGroups
  - DescribeServices
  - DescribeSubnets
  - DescribeTasks
  - DescribeVpcs
  - GetCallerIdentity
  - GetRole
  - ListTaskDefinitions
  - ListTasks
  - RegisterTaskDefinition
  - UpdateService
  - Attente

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Création d'une API REST avec intégration de proxy de fonctions

L'exemple de code suivant illustre comment :

- Création d'un rôle IAM pour l'exécution de Lambda
- Création et déploiement d'une fonction Lambda
- Création d'une API REST
- Configurer l'intégration du proxy Lambda
- Déploiement et test de l'API
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

set -euo pipefail

# Simple API Gateway Lambda Integration Script
# This script creates a REST API with Lambda proxy integration

# Generate random identifiers
FUNCTION_NAME="GetStartedLambdaProxyIntegration-$(openssl rand -hex 4)"
ROLE_NAME="GetStartedLambdaBasicExecutionRole-$(openssl rand -hex 4)"
API_NAME="LambdaProxyAPI-$(openssl rand -hex 4)"

# Get AWS account info
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
REGION=$(aws configure get region || echo "us-east-1")

# Validate inputs
if [[ -z "$ACCOUNT_ID" ]] || [[ -z "$REGION" ]]; then
```

```
    echo "Error: Failed to retrieve AWS account information" >&2
    exit 1
fi

echo "Creating Lambda function code..."

# Create Lambda function code with input validation
cat > lambda_function.py << 'EOF'
import json
import logging

logger = logging.getLogger()
logger.setLevel(logging.INFO)

def lambda_handler(event, context):
    try:
        logger.info("Received event: %s", json.dumps(event))

        greeter = 'World'

        # Safely retrieve greeter from query string parameters
        query_params = event.get('queryStringParameters') or {}
        if isinstance(query_params, dict) and 'greeter' in query_params:
            greeter_value = query_params.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
                greeter = greeter_value

        # Safely retrieve greeter from multi-value headers
        multi_headers = event.get('multiValueHeaders') or {}
        if isinstance(multi_headers, dict) and 'greeter' in multi_headers:
            greeter_list = multi_headers.get('greeter', [])
            if isinstance(greeter_list, list) and greeter_list:
                greeter = " and ".join(str(g) for g in greeter_list if g)

        # Safely retrieve greeter from headers
        headers = event.get('headers') or {}
        if isinstance(headers, dict) and 'greeter' in headers:
            greeter_value = headers.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
                greeter = greeter_value

        # Safely retrieve greeter from body
        body = event.get('body')
        if body and isinstance(body, str):
```

```
            try:
                body_dict = json.loads(body)
                if isinstance(body_dict, dict) and 'greeter' in body_dict:
                    greeter_value = body_dict.get('greeter')
                    if isinstance(greeter_value, str) and greeter_value:
                        greeter = greeter_value
            except (json.JSONDecodeError, ValueError) as e:
                logger.warning("Failed to parse body: %s", str(e))

        # Sanitize greeter to prevent injection
        greeter = greeter.replace('"', '\\"').replace("'", "\\'")

        response = {
            "statusCode": 200,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"message": f"Hello, {greeter}!"})
        }

        logger.info("Response: %s", json.dumps(response))
        return response

    except Exception as e:
        logger.error("Unexpected error: %s", str(e), exc_info=True)
        return {
            "statusCode": 500,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"error": "Internal server error"})
        }
EOF

# Create deployment package
zip -q function.zip lambda_function.py || {
    echo "Error: Failed to create function.zip" >&2
    exit 1
}

echo "Creating IAM role..."

# Create IAM trust policy
cat > trust-policy.json << 'EOF'
```

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role with error handling
aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document file://trust-policy.json \
    --description "Temporary role for Lambda execution" || {
    echo "Error: Failed to create IAM role" >&2
    exit 1
}

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=apigateway-lambda-integration

# Attach execution policy
aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || {
    echo "Error: Failed to attach IAM policy" >&2
    exit 1
}

# Wait for role propagation
sleep 15

echo "Creating Lambda function..."

# Create Lambda function with Python 3.11 (more recent runtime)
aws lambda create-function \
    --function-name "$FUNCTION_NAME" \
    --runtime python3.11 \
```

```
    --role "arn:aws:iam::$ACCOUNT_ID:role/$ROLE_NAME" \
    --handler lambda_function.lambda_handler \
    --zip-file fileb://function.zip \
    --timeout 30 \
    --memory-size 128 \
    --environment "Variables={LOG_LEVEL=INFO}" \
    --tags project=doc-smith,tutorial=apigateway-lambda-integration || {
    echo "Error: Failed to create Lambda function" >&2
    exit 1
}

echo "Creating API Gateway..."

# Create REST API with minimum logging
API_RESPONSE=$(aws apigateway create-rest-api \
    --name "$API_NAME" \
    --endpoint-configuration types=REGIONAL \
    --description "API for Lambda proxy integration tutorial" \
    --tags project=doc-smith,tutorial=apigateway-lambda-integration \
    --output json)

API_ID=$(echo "$API_RESPONSE" | grep -o '"id": "[^"]*"' | head -1 | cut -d'"' -
f4)

if [[ -z "$API_ID" ]]; then
    echo "Error: Failed to create API Gateway" >&2
    exit 1
fi

# Get root resource ID
ROOT_RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 'items[?path==`/`].id' --output text)

# Create helloworld resource
aws apigateway create-resource \
    --rest-api-id "$API_ID" \
    --parent-id "$ROOT_RESOURCE_ID" \
    --path-part helloworld || {
    echo "Error: Failed to create resource" >&2
    exit 1
}

# Get resource ID
```

```
RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 "items[?pathPart=='helloworld'].id" --output text)

# Create ANY method with no authorization (intentional for tutorial)
aws apigateway put-method \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
    --http-method ANY \
    --authorization-type NONE || {
    echo "Error: Failed to create method" >&2
    exit 1
}

# Set up Lambda proxy integration
LAMBDA_URI="arn:aws:apigateway:$REGION:lambda:path/2015-03-31/functions/
arn:aws:lambda:$REGION:$ACCOUNT_ID:function:$FUNCTION_NAME/invocations"

aws apigateway put-integration \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
    --http-method ANY \
    --type AWS_PROXY \
    --integration-http-method POST \
    --uri "$LAMBDA_URI" || {
    echo "Error: Failed to create integration" >&2
    exit 1
}

# Grant API Gateway permission to invoke Lambda
STATEMENT_ID="apigateway-invoke-$(openssl rand -hex 4)"
SOURCE_ARN="arn:aws:execute-api:$REGION:$ACCOUNT_ID:$API_ID/*/*"

aws lambda add-permission \
    --function-name "$FUNCTION_NAME" \
    --statement-id "$STATEMENT_ID" \
    --action lambda:InvokeFunction \
    --principal apigateway.amazonaws.com \
    --source-arn "$SOURCE_ARN" || {
    echo "Error: Failed to add Lambda permission" >&2
    exit 1
}

# Deploy API
aws apigateway create-deployment \
```

```
    --rest-api-id "$API_ID" \
    --stage-name test \
    --description "Test deployment" || {
    echo "Error: Failed to deploy API" >&2
    exit 1
}

echo "Testing API..."

# Test the API
INVOKE_URL="https://$API_ID.execute-api.$REGION.amazonaws.com/test/helloworld"

echo "API URL: $INVOKE_URL"

# Test with query parameter (with proper URL encoding)
echo "Testing with query parameter:"
curl -s -X GET "$INVOKE_URL?greeter=John" | jq . 2>/dev/null || curl -s -X GET
 "$INVOKE_URL?greeter=John"
echo ""

# Test with header
echo "Testing with header:"
curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -H 'greeter: John' | jq . 2>/dev/null || curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -H 'greeter: John'
echo ""

# Test with body
echo "Testing with POST body:"
curl -s -X POST "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}' | jq . 2>/dev/null || curl -s -X POST "$INVOKE_URL"
 \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}'
echo ""

echo "Tutorial completed! API is available at: $INVOKE_URL"

# Cleanup
echo "Cleaning up resources..."
```

```
# Delete API
aws apigateway delete-rest-api --rest-api-id "$API_ID" || echo "Warning: Failed
 to delete API" >&2

# Delete Lambda function
aws lambda delete-function --function-name "$FUNCTION_NAME" || echo "Warning:
 Failed to delete Lambda function" >&2

# Detach policy and delete role
aws iam detach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || echo "Warning: Failed to detach policy" >&2

aws iam delete-role --role-name "$ROLE_NAME" || echo "Warning: Failed to delete
 role" >&2

# Clean up local files securely
rm -f lambda_function.py function.zip trust-policy.json

echo "Cleanup completed!"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AddPermission
  - AttachRolePolicy
  - CreateDeployment
  - CreateFunction
  - CreateResource
  - CreateRestApi
  - CreateRole
  - DeleteFunction
  - DeleteRestApi
  - DeleteRole
  - DetachRolePolicy
  - GetCallerIdentity
  - GetResources

- GetRestApis
- PutIntegration
- PutMethod

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Création d'un service de conteneur pour les instances de machines virtuelles

L'exemple de code suivant illustre comment :

- Création d'un cluster ECS
- Création et surveillance d'un service
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# ECS EC2 Launch Type Tutorial Script - UPDATED VERSION
# This script demonstrates creating an ECS cluster, launching a container
 instance,
# registering a task definition, and creating a service using the EC2 launch
 type.
# Updated to match the tutorial draft with nginx web server and service creation.
#
# - UPDATED: Changed from sleep task to nginx web server with service

set -e  # Exit on any error
```

```
# Configuration
SCRIPT_NAME="ecs-ec2-tutorial"
LOG_FILE="${SCRIPT_NAME}-$(date +%Y%m%d-%H%M%S).log"
CLUSTER_NAME="tutorial-cluster-$(openssl rand -hex 4)"
TASK_FAMILY="nginx-task-$(openssl rand -hex 4)"
SERVICE_NAME="nginx-service-$(openssl rand -hex 4)"
KEY_PAIR_NAME="ecs-tutorial-key-$(openssl rand -hex 4)"
SECURITY_GROUP_NAME="ecs-tutorial-sg-$(openssl rand -hex 4)"

# Tags
PROJECT_TAG="doc-smith"
TUTORIAL_TAG="ecs-ec2"

# Get current AWS region dynamically
AWS_REGION=$(aws configure get region || echo "us-east-1")

# Resource tracking arrays
CREATED_RESOURCES=()
CLEANUP_ORDER=()

# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
}

# Error handling function
handle_error() {
    local exit_code=$?
    log "ERROR: Script failed with exit code $exit_code"
    log "ERROR: Last command: $BASH_COMMAND"

    echo ""
    echo "========================================="
    echo "ERROR OCCURRED - ATTEMPTING CLEANUP"
    echo "========================================="
    echo "Resources created before error:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done

    cleanup_resources
    exit $exit_code
}
```

```
# Set error trap
trap handle_error ERR

# FIXED: Enhanced cleanup function with proper error handling and logging
cleanup_resources() {
    log "Starting cleanup process..."
    local cleanup_errors=0

    # Delete service first (this will stop tasks automatically)
    if [[ -n "${SERVICE_ARN:-}" ]]; then
        log "Updating service to desired count 0: $SERVICE_NAME"
        if ! aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" --desired-count 0 2>>"$LOG_FILE"; then
            log "WARNING: Failed to update service desired count to 0"
            ((cleanup_errors++))
        else
            log "Waiting for service tasks to stop..."
            sleep 30  # Give time for tasks to stop
        fi

        log "Deleting service: $SERVICE_NAME"
        if ! aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to delete service $SERVICE_NAME"
            ((cleanup_errors++))
        fi
    fi

    # Stop and delete any remaining tasks
    if [[ -n "${TASK_ARN:-}" ]]; then
        log "Stopping task: $TASK_ARN"
        if ! aws ecs stop-task --cluster "$CLUSTER_NAME" --task "$TASK_ARN" --
reason "Tutorial cleanup" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to stop task $TASK_ARN"
            ((cleanup_errors++))
        else
            log "Waiting for task to stop..."
            if ! aws ecs wait tasks-stopped --cluster "$CLUSTER_NAME" --tasks
 "$TASK_ARN" 2>>"$LOG_FILE"; then
                log "WARNING: Task stop wait failed for $TASK_ARN"
                ((cleanup_errors++))
            fi
        fi
    fi
```

```
   # Deregister task definition
   if [[ -n "${TASK_DEFINITION_ARN:-}" ]]; then
       log "Deregistering task definition: $TASK_DEFINITION_ARN"
       if ! aws ecs deregister-task-definition --task-definition
"$TASK_DEFINITION_ARN" 2>>"$LOG_FILE"; then
           log "WARNING: Failed to deregister task definition
$TASK_DEFINITION_ARN"
           ((cleanup_errors++))
       fi
   fi

   # Terminate EC2 instance
   if [[ -n "${INSTANCE_ID:-}" ]]; then
       log "Terminating EC2 instance: $INSTANCE_ID"
       if ! aws ec2 terminate-instances --instance-ids "$INSTANCE_ID"
2>>"$LOG_FILE"; then
           log "WARNING: Failed to terminate instance $INSTANCE_ID"
           ((cleanup_errors++))
       else
           log "Waiting for instance to terminate..."
           if ! aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"
2>>"$LOG_FILE"; then
               log "WARNING: Instance termination wait failed for $INSTANCE_ID"
               ((cleanup_errors++))
           fi
       fi
   fi

   # Delete security group with retry logic
   if [[ -n "${SECURITY_GROUP_ID:-}" ]]; then
       log "Deleting security group: $SECURITY_GROUP_ID"
       local retry_count=0
       local max_retries=3

       while [[ $retry_count -lt $max_retries ]]; do
           if aws ec2 delete-security-group --group-id "$SECURITY_GROUP_ID"
2>>"$LOG_FILE"; then
               log "Successfully deleted security group"
               break
           else
               ((retry_count++))
               if [[ $retry_count -lt $max_retries ]]; then
```

```
                    log "Retry $retry_count/$max_retries: Waiting 10 seconds
 before retrying security group deletion..."
                    sleep 10
                else
                    log "ERROR: Failed to delete security group after
 $max_retries attempts"
                    ((cleanup_errors++))
                fi
            fi
        done
    fi

    # Delete key pair
    if [[ -n "${KEY_PAIR_NAME:-}" ]]; then
        log "Deleting key pair: $KEY_PAIR_NAME"
        if ! aws ec2 delete-key-pair --key-name "$KEY_PAIR_NAME" 2>>"$LOG_FILE";
 then
            log "WARNING: Failed to delete key pair $KEY_PAIR_NAME"
            ((cleanup_errors++))
        fi
        rm -f "${KEY_PAIR_NAME}.pem" 2>>"$LOG_FILE" || log "WARNING: Failed to
 remove local key file"
    fi

    # Delete ECS cluster
    if [[ -n "${CLUSTER_NAME:-}" ]]; then
        log "Deleting ECS cluster: $CLUSTER_NAME"
        if ! aws ecs delete-cluster --cluster "$CLUSTER_NAME" 2>>"$LOG_FILE";
 then
            log "WARNING: Failed to delete cluster $CLUSTER_NAME"
            ((cleanup_errors++))
        fi
    fi

    if [[ $cleanup_errors -eq 0 ]]; then
        log "Cleanup completed successfully"
    else
        log "Cleanup completed with $cleanup_errors warnings/errors. Check log
 file for details."
    fi
}

# Function to check prerequisites
check_prerequisites() {
```

```
    log "Checking prerequisites..."

    # Check AWS CLI
    if ! command -v aws &> /dev/null; then
        log "ERROR: AWS CLI is not installed"
        exit 1
    fi

    # Check AWS credentials
    if ! aws sts get-caller-identity &> /dev/null; then
        log "ERROR: AWS credentials not configured"
        exit 1
    fi

    # Get caller identity
    CALLER_IDENTITY=$(aws sts get-caller-identity --output text --query
 'Account')
    log "AWS Account: $CALLER_IDENTITY"
    log "AWS Region: $AWS_REGION"

    # Check for default VPC
    DEFAULT_VPC=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true"
 --query 'Vpcs[0].VpcId' --output text)
    if [[ "$DEFAULT_VPC" == "None" ]]; then
        log "ERROR: No default VPC found. Please create a VPC first."
        exit 1
    fi
    log "Using default VPC: $DEFAULT_VPC"

    # Get default subnet
    DEFAULT_SUBNET=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=
$DEFAULT_VPC" "Name=default-for-az,Values=true" --query 'Subnets[0].SubnetId' --
output text)
    if [[ "$DEFAULT_SUBNET" == "None" ]]; then
        log "ERROR: No default subnet found"
        exit 1
    fi
    log "Using default subnet: $DEFAULT_SUBNET"

    log "Prerequisites check completed successfully"
}

# Function to create ECS cluster
create_cluster() {
```

```
    log "Creating ECS cluster: $CLUSTER_NAME"

    CLUSTER_ARN=$(aws ecs create-cluster --cluster-name "$CLUSTER_NAME" --
tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG --query
 'cluster.clusterArn' --output text)

    if [[ -z "$CLUSTER_ARN" ]]; then
        log "ERROR: Failed to create cluster"
        exit 1
    fi

    log "Created cluster: $CLUSTER_ARN"
    CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")
}

# Function to create key pair
create_key_pair() {
    log "Creating EC2 key pair: $KEY_PAIR_NAME"

    # FIXED: Set secure umask before key creation
    umask 077
    aws ec2 create-key-pair --key-name "$KEY_PAIR_NAME" --query 'KeyMaterial' --
output text > "${KEY_PAIR_NAME}.pem"
    chmod 400 "${KEY_PAIR_NAME}.pem"
    umask 022  # Reset umask

    log "Created key pair: $KEY_PAIR_NAME"
    CREATED_RESOURCES+=("EC2 Key Pair: $KEY_PAIR_NAME")

    aws ec2 create-tags --resources "$KEY_PAIR_NAME" --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG 2>>"$LOG_FILE" || log "WARNING:
 Failed to tag key pair"
}

# Function to create security group
create_security_group() {
    log "Creating security group: $SECURITY_GROUP_NAME"

    SECURITY_GROUP_ID=$(aws ec2 create-security-group \
        --group-name "$SECURITY_GROUP_NAME" \
        --description "ECS tutorial security group" \
        --vpc-id "$DEFAULT_VPC" \
```

```
        --tag-specifications "ResourceType=security-
group,Tags=[{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=$TUTORIAL_TAG}]"
 \
        --query 'GroupId' --output text)

    if [[ -z "$SECURITY_GROUP_ID" ]]; then
        log "ERROR: Failed to create security group"
        exit 1
    fi

    # Add HTTP access rule for nginx web server with restricted CIDR
    # SECURITY FIX: Restrict access to specific CIDR if available, otherwise
 document the risk
    log "WARNING: Security group allows HTTP (port 80) from 0.0.0.0/0 - restrict
 this in production"
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr "0.0.0.0/0"

    log "Created security group: $SECURITY_GROUP_ID"
    log "Added HTTP access on port 80"
    CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")
}

# Function to get ECS optimized AMI
get_ecs_ami() {
    log "Getting ECS-optimized AMI ID..."

    ECS_AMI_ID=$(aws ssm get-parameters \
        --names /aws/service/ecs/optimized-ami/amazon-linux-2/recommended \
        --query 'Parameters[0].Value' --output text | jq -r '.image_id')

    if [[ -z "$ECS_AMI_ID" ]]; then
        log "ERROR: Failed to get ECS-optimized AMI ID"
        exit 1
    fi

    log "ECS-optimized AMI ID: $ECS_AMI_ID"
}

# Function to create IAM role for ECS instance (if it doesn't exist)
ensure_ecs_instance_role() {
```

```
    log "Checking for ecsInstanceRole..."

    if ! aws iam get-role --role-name ecsInstanceRole &> /dev/null; then
        log "Creating ecsInstanceRole..."

        # Create trust policy
        cat > ecs-instance-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

        # SECURITY FIX: Validate JSON before using
        if ! jq empty ecs-instance-trust-policy.json 2>/dev/null; then
            log "ERROR: Invalid JSON in trust policy"
            rm -f ecs-instance-trust-policy.json
            exit 1
        fi

        # Create role
        aws iam create-role \
            --role-name ecsInstanceRole \
            --assume-role-policy-document file://ecs-instance-trust-policy.json

        aws iam tag-role --role-name ecsInstanceRole --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG

        # Attach managed policy
        aws iam attach-role-policy \
            --role-name ecsInstanceRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role

        # Create instance profile
        aws iam create-instance-profile --instance-profile-name ecsInstanceRole
```

```
        # Add role to instance profile
        aws iam add-role-to-instance-profile \
            --instance-profile-name ecsInstanceRole \
            --role-name ecsInstanceRole

        # FIXED: Enhanced wait for role to be ready
        log "Waiting for IAM role to be ready..."
        aws iam wait role-exists --role-name ecsInstanceRole
        sleep 30  # Additional buffer for eventual consistency

        rm -f ecs-instance-trust-policy.json
        log "Created ecsInstanceRole"
        CREATED_RESOURCES+=("IAM Role: ecsInstanceRole")
    else
        log "ecsInstanceRole already exists"
    fi
}

# Function to launch container instance
launch_container_instance() {
    log "Launching ECS container instance..."

    # Create user data script
    cat > ecs-user-data.sh << EOF
#!/bin/bash
echo ECS_CLUSTER=$CLUSTER_NAME >> /etc/ecs/ecs.config
EOF

    # SECURITY FIX: Validate user data script before use
    if ! bash -n ecs-user-data.sh 2>/dev/null; then
        log "ERROR: Invalid user data script"
        rm -f ecs-user-data.sh
        exit 1
    fi

    INSTANCE_ID=$(aws ec2 run-instances \
        --image-id "$ECS_AMI_ID" \
        --instance-type t3.micro \
        --key-name "$KEY_PAIR_NAME" \
        --security-group-ids "$SECURITY_GROUP_ID" \
        --subnet-id "$DEFAULT_SUBNET" \
        --iam-instance-profile Name=ecsInstanceRole \
        --user-data file://ecs-user-data.sh \
```

```
        --tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=ecs-
tutorial-instance},{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=
$TUTORIAL_TAG}]" \
        --monitoring Enabled=false \
        --metadata-options HttpTokens=required,HttpPutResponseHopLimit=1 \
        --query 'Instances[0].InstanceId' --output text)

    if [[ -z "$INSTANCE_ID" ]]; then
        log "ERROR: Failed to launch EC2 instance"
        rm -f ecs-user-data.sh
        exit 1
    fi

    log "Launched EC2 instance: $INSTANCE_ID"
    log "Instance metadata options: IMDSv2 enforced with hop limit 1"
    CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")

    # Wait for instance to be running
    log "Waiting for instance to be running..."
    aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

    # Wait for ECS agent to register
    log "Waiting for ECS agent to register with cluster..."
    local max_attempts=30
    local attempt=0

    while [[ $attempt -lt $max_attempts ]]; do
        CONTAINER_INSTANCES=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns' --output text)
        if [[ -n "$CONTAINER_INSTANCES" && "$CONTAINER_INSTANCES" != "None" ]];
 then
            log "Container instance registered successfully"
            break
        fi

        attempt=$((attempt + 1))
        log "Waiting for container instance registration... (attempt $attempt/
$max_attempts)"
        sleep 10
    done

    if [[ $attempt -eq $max_attempts ]]; then
        log "ERROR: Container instance failed to register within expected time"
        rm -f ecs-user-data.sh
```

```
        exit 1
    fi

    rm -f ecs-user-data.sh
}

# Function to register task definition
register_task_definition() {
    log "Creating task definition..."

    # Create nginx task definition JSON matching the tutorial
    cat > task-definition.json << 'EOF'
{
    "family": "TASK_FAMILY_PLACEHOLDER",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "cpu": 256,
            "memory": 512,
            "essential": true,
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/nginx-task",
                    "awslogs-region": "REGION_PLACEHOLDER",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ],
    "requiresCompatibilities": ["EC2"],
    "networkMode": "bridge",
    "tags": [
        {
            "key": "project",
            "value": "PROJECT_TAG_PLACEHOLDER"
```

```
        },
        {
            "key": "tutorial",
            "value": "TUTORIAL_TAG_PLACEHOLDER"
        }
    ]
}
EOF

    # Replace placeholders securely
    sed -i "s|TASK_FAMILY_PLACEHOLDER|$TASK_FAMILY|g" task-definition.json
    sed -i "s|REGION_PLACEHOLDER|$AWS_REGION|g" task-definition.json
    sed -i "s|PROJECT_TAG_PLACEHOLDER|$PROJECT_TAG|g" task-definition.json
    sed -i "s|TUTORIAL_TAG_PLACEHOLDER|$TUTORIAL_TAG|g" task-definition.json

    # FIXED: Validate JSON before registration
    if ! jq empty task-definition.json 2>/dev/null; then
        log "ERROR: Invalid JSON in task definition"
        rm -f task-definition.json
        exit 1
    fi

    TASK_DEFINITION_ARN=$(aws ecs register-task-definition \
        --cli-input-json file://task-definition.json \
        --query 'taskDefinition.taskDefinitionArn' --output text)

    if [[ -z "$TASK_DEFINITION_ARN" ]]; then
        log "ERROR: Failed to register task definition"
        rm -f task-definition.json
        exit 1
    fi

    log "Registered task definition: $TASK_DEFINITION_ARN"
    log "Task definition includes CloudWatch Logs configuration for monitoring"
    CREATED_RESOURCES+=("Task Definition: $TASK_DEFINITION_ARN")

    rm -f task-definition.json
}

# Function to create service
create_service() {
    log "Creating ECS service..."

    SERVICE_ARN=$(aws ecs create-service \
```

```
        --cluster "$CLUSTER_NAME" \
        --service-name "$SERVICE_NAME" \
        --task-definition "$TASK_FAMILY" \
        --desired-count 1 \
        --tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG \
        --query 'service.serviceArn' --output text)

    if [[ -z "$SERVICE_ARN" ]]; then
        log "ERROR: Failed to create service"
        exit 1
    fi

    log "Created service: $SERVICE_ARN"
    CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")

    # Wait for service to be stable
    log "Waiting for service to be stable..."
    aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$SERVICE_NAME"

    log "Service is now stable and running"

    # Get the task ARN for monitoring
    TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query 'taskArns[0]' --output text)
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Service task: $TASK_ARN"
        CREATED_RESOURCES+=("ECS Task: $TASK_ARN")
    fi
}

# Function to demonstrate monitoring and testing
demonstrate_monitoring() {
    log "Demonstrating monitoring capabilities..."

    # List services
    log "Listing services in cluster:"
    aws ecs list-services --cluster "$CLUSTER_NAME" --output table

    # Describe service
    log "Service details:"
    aws ecs describe-services --cluster "$CLUSTER_NAME" --
services "$SERVICE_NAME" --output table --query 'services[0].
{ServiceName:serviceName,Status:status,RunningCount:runningCount,DesiredCount:desiredCoun
```

```
    # List tasks
    log "Listing tasks in service:"
    aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name "$SERVICE_NAME"
 --output table

    # Describe task
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Task details:"
        aws ecs describe-tasks --cluster "$CLUSTER_NAME"
 --tasks "$TASK_ARN" --output table --query 'tasks[0].
{TaskArn:taskArn,LastStatus:lastStatus,DesiredStatus:desiredStatus,CreatedAt:createdAt}'
    fi

    # List container instances
    log "Container instances in cluster:"
    aws ecs list-container-instances --cluster "$CLUSTER_NAME" --output table

    # Describe container instance
    CONTAINER_INSTANCE_ARN=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns[0]' --output text)
    if [[ -n "$CONTAINER_INSTANCE_ARN" && "$CONTAINER_INSTANCE_ARN" != "None" ]];
 then
        log "Container instance details:"
        aws ecs describe-container-instances --cluster "$CLUSTER_NAME"
 --container-instances "$CONTAINER_INSTANCE_ARN" --output table --query
 'containerInstances[0].
{Arn:containerInstanceArn,Status:status,RunningTasks:runningTasksCount,PendingTasks:pendi
    fi

    # Test the nginx web server
    log "Testing nginx web server..."
    PUBLIC_IP=$(aws ec2 describe-instances --instance-ids "$INSTANCE_ID" --query
 'Reservations[0].Instances[0].PublicIpAddress' --output text)

    if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
        log "Container instance public IP: $PUBLIC_IP"
        log "Testing HTTP connection to nginx..."

        # Wait a moment for nginx to be fully ready
        sleep 10

        if curl -s --connect-timeout 10 "http://$PUBLIC_IP" | grep -q "Welcome to
 nginx"; then
```

```
            log "SUCCESS: Nginx web server is responding correctly"
            echo ""
            echo "=========================================="
            echo "WEB SERVER TEST SUCCESSFUL"
            echo "=========================================="
            echo "You can access your nginx web server at: http://$PUBLIC_IP"
            echo "The nginx welcome page should be visible in your browser."
        else
            log "WARNING: Nginx web server may not be fully ready yet. Try
 accessing http://$PUBLIC_IP in a few minutes."
        fi
    else
        log "WARNING: Could not retrieve public IP address"
    fi
}

# Main execution
main() {
    log "Starting ECS EC2 Launch Type Tutorial (UPDATED VERSION)"
    log "Log file: $LOG_FILE"
    log "Security improvements: IMDSv2 enforced, JSON validation, input
 sanitization, CloudWatch Logs configured"

    check_prerequisites
    create_cluster
    create_key_pair
    create_security_group
    get_ecs_ami
    ensure_ecs_instance_role
    launch_container_instance
    register_task_definition
    create_service
    demonstrate_monitoring

    log "Tutorial completed successfully!"

    echo ""
    echo "=========================================="
    echo "TUTORIAL COMPLETED SUCCESSFULLY"
    echo "=========================================="
    echo "Resources created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
```

```
    echo ""
    echo "The nginx service will continue running and maintain the desired task
 count."
    echo "You can monitor the service status using:"
    echo "  aws ecs describe-services --cluster $CLUSTER_NAME --services
 $SERVICE_NAME"
    echo ""
    if [[ -n "${PUBLIC_IP:-}" ]]; then
        echo "Access your web server at: http://$PUBLIC_IP"
        echo ""
    fi
    echo "==========================================="
    echo "CLEANUP CONFIRMATION"
    echo "==========================================="
    log "Auto-confirming cleanup - proceeding with resource cleanup"
    cleanup_resources
    log "All resources have been cleaned up"
}

# Run main function
main "$@"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateInstanceProfile
  - CreateKeyPair
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DeleteService

- DeregisterTaskDefinition
- DescribeContainerInstances
- DescribeInstances
- DescribeServices
- DescribeSubnets
- DescribeTasks
- DescribeVpcs
- GetCallerIdentity
- GetParameters
- GetRole
- ListContainerInstances
- ListServices
- ListTasks
- RegisterTaskDefinition
- RunInstances
- StopTask
- TerminateInstances
- UpdateService
- Attendre (Amazon EC2)
- Attendre (Amazon ECS)
- Attendre (IAM)

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Création d'un espace de travail de surveillance géré

L'exemple de code suivant illustre comment :

- Créez un rôle IAM pour votre espace de travail
- Création d'un espace de travail Grafana
- Configuration de l'authentification

- Configurer les paramètres facultatifs
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Amazon Managed Grafana Workspace Creation Script
# This script creates an Amazon Managed Grafana workspace and configures it

# Set up logging
LOG_FILE="grafana-workspace-creation.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Managed Grafana workspace creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -i "error\|exception\|fail" > /dev/null; then
        echo "ERROR: Command '$cmd' failed with output:"
        echo "$output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Attempting to clean up resources..."
```

```
    if [ -n "$WORKSPACE_ID" ]; then
        echo "Deleting workspace $WORKSPACE_ID..."
        aws grafana delete-workspace --workspace-id "$WORKSPACE_ID"
    fi

    if [ -n "$ROLE_NAME" ]; then
        echo "Detaching policies from role $ROLE_NAME..."
        if [ -n "$POLICY_ARN" ]; then
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$POLICY_ARN"
        fi

        echo "Deleting role $ROLE_NAME..."
        aws iam delete-role --role-name "$ROLE_NAME"
    fi

    if [ -n "$POLICY_ARN" ]; then
        echo "Deleting policy..."
        aws iam delete-policy --policy-arn "$POLICY_ARN"
    fi

    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json

    echo "Cleanup completed. See $LOG_FILE for details."
}

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
WORKSPACE_NAME="GrafanaWorkspace-${RANDOM_ID}"
ROLE_NAME="GrafanaWorkspaceRole-${RANDOM_ID}"

echo "Using workspace name: $WORKSPACE_NAME"
echo "Using role name: $ROLE_NAME"

# Step 1: Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
check_error "$ACCOUNT_ID" "get-caller-identity"
echo "AWS Account ID: $ACCOUNT_ID"

# Step 2: Create IAM role for Grafana workspace
echo "Creating IAM role for Grafana workspace..."
```

```
# Create trust policy document
cat > trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "grafana.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role
ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
  --assume-role-policy-document file://trust-policy.json \
  --description "Role for Amazon Managed Grafana workspace")

check_error "$ROLE_OUTPUT" "create-role"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-grafana-gs
echo "IAM role created successfully"

# Extract role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "Role ARN: $ROLE_ARN"

# Attach policies to the role
echo "Attaching policies to the role..."

# CloudWatch policy
cat > cloudwatch-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:DescribeAlarmsForMetric",
```

```
        "cloudwatch:DescribeAlarmHistory",
        "cloudwatch:DescribeAlarms",
        "cloudwatch:ListMetrics",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:GetMetricData"
      ],
      "Resource": "*"
    }
  ]
}
EOF

POLICY_OUTPUT=$(aws iam create-policy \
  --policy-name "GrafanaCloudWatchPolicy-${RANDOM_ID}" \
  --policy-document file://cloudwatch-policy.json)

check_error "$POLICY_OUTPUT" "create-policy"

POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "CloudWatch policy ARN: $POLICY_ARN"

ATTACH_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-arn "$POLICY_ARN")

check_error "$ATTACH_OUTPUT" "attach-role-policy"
echo "CloudWatch policy attached to role"

# Step 3: Create the Grafana workspace
echo "Creating Amazon Managed Grafana workspace..."
WORKSPACE_OUTPUT=$(aws grafana create-workspace \
  --workspace-name "$WORKSPACE_NAME" \
  --authentication-providers "SAML" \
  --permission-type "CUSTOMER_MANAGED" \
  --account-access-type "CURRENT_ACCOUNT" \
  --workspace-role-arn "$ROLE_ARN" \
  --workspace-data-sources "CLOUDWATCH" "PROMETHEUS" "XRAY" \
  --grafana-version "10.4" \
  --tags Environment=Development,project=doc-smith,tutorial=amazon-managed-
grafana-gs)

check_error "$WORKSPACE_OUTPUT" "create-workspace"

echo "Workspace creation initiated:"
```

```
echo "$WORKSPACE_OUTPUT"

# Extract workspace ID
WORKSPACE_ID=$(echo "$WORKSPACE_OUTPUT" | grep -o '"id": "[^"]*' | cut -d'"' -f4)

if [ -z "$WORKSPACE_ID" ]; then
    echo "ERROR: Failed to extract workspace ID from output"
    exit 1
fi

echo "Workspace ID: $WORKSPACE_ID"

# Step 4: Wait for workspace to become active
echo "Waiting for workspace to become active. This may take several minutes..."
ACTIVE=false
MAX_ATTEMPTS=30
ATTEMPT=0

while [ $ACTIVE = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
    ATTEMPT=$((ATTEMPT+1))
    echo "Checking workspace status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."

    DESCRIBE_OUTPUT=$(aws grafana describe-workspace --workspace-id
 "$WORKSPACE_ID")
    check_error "$DESCRIBE_OUTPUT" "describe-workspace"

    STATUS=$(echo "$DESCRIBE_OUTPUT" | grep -o '"status": "[^"]*' | cut -d'"' -
f4)
    echo "Current status: $STATUS"

    if [ "$STATUS" = "ACTIVE" ]; then
        ACTIVE=true
        echo "Workspace is now ACTIVE"
    elif [ "$STATUS" = "FAILED" ]; then
        echo "ERROR: Workspace creation failed"
        cleanup_on_error
        exit 1
    else
        echo "Workspace is still being created. Waiting 30 seconds..."
        sleep 30
    fi
done

if [ $ACTIVE = false ]; then
```

```
    echo "ERROR: Workspace did not become active within the expected time"
    cleanup_on_error
    exit 1
fi

# Extract workspace endpoint URL
WORKSPACE_URL=$(echo "$DESCRIBE_OUTPUT" | grep -o '"endpoint": "[^"]*' | cut -
d'"' -f4)
echo "Workspace URL: https://$WORKSPACE_URL"

# Step 5: Display workspace information
echo ""
echo "==========================================="
echo "WORKSPACE INFORMATION"
echo "==========================================="
echo "Workspace ID: $WORKSPACE_ID"
echo "Workspace URL: https://$WORKSPACE_URL"
echo "Workspace Name: $WORKSPACE_NAME"
echo "IAM Role: $ROLE_NAME"
echo ""
echo "Note: Since SAML authentication is used, you need to configure SAML
 settings"
echo "using the AWS Management Console or the update-workspace-authentication
 command."
echo "==========================================="

# Step 6: Prompt for cleanup
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Resources created:"
echo "- Amazon Managed Grafana workspace: $WORKSPACE_ID"
echo "- IAM Role: $ROLE_NAME"
echo "- IAM Policy: GrafanaCloudWatchPolicy-${RANDOM_ID}"
echo ""
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy] ]]; then
    echo "Cleaning up resources..."

    echo "Deleting workspace $WORKSPACE_ID..."
    DELETE_OUTPUT=$(aws grafana delete-workspace --workspace-id "$WORKSPACE_ID")
```

```
    check_error "$DELETE_OUTPUT" "delete-workspace"

    echo "Waiting for workspace to be deleted..."
    DELETED=false
    ATTEMPT=0

    while [ $DELETED = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
        ATTEMPT=$((ATTEMPT+1))
        echo "Checking deletion status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."

        if aws grafana describe-workspace --workspace-id "$WORKSPACE_ID" 2>&1 |
grep -i "not found\|does not exist" > /dev/null; then
            DELETED=true
            echo "Workspace has been deleted"
        else
            echo "Workspace is still being deleted. Waiting 30 seconds..."
            sleep 30
        fi
    done

    if [ $DELETED = false ]; then
        echo "WARNING: Workspace deletion is taking longer than expected. It may
still be in progress."
    fi

    # Detach policy from role
    echo "Detaching policy from role..."
    aws iam detach-role-policy \
      --role-name "$ROLE_NAME" \
      --policy-arn "$POLICY_ARN"

    # Delete policy
    echo "Deleting IAM policy..."
    aws iam delete-policy \
      --policy-arn "$POLICY_ARN"

    # Delete role
    echo "Deleting IAM role..."
    aws iam delete-role \
      --role-name "$ROLE_NAME"

    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json
```

```
    echo "Cleanup completed"
else
    echo "Skipping cleanup. Resources will remain in your AWS account."
fi

echo "Script completed at $(date)"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreatePolicy
  - CreateRole
  - CreateWorkspace
  - DeletePolicy
  - DeleteRole
  - DeleteWorkspace
  - DescribeWorkspace
  - DetachRolePolicy
  - GetCallerIdentity

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Création d'un tableau de bord de surveillance avec le nom de la fonction comme variable

L'exemple de code suivant illustre comment :

- Création d'un CloudWatch tableau de bord
- Ajouter des widgets de métriques Lambda avec une variable de nom de fonction
- Vérifiez le tableau de bord
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Script to create a CloudWatch dashboard with Lambda function name as a variable
# This script creates a CloudWatch dashboard that allows you to switch between
 different Lambda functions

# Set up logging with secure permissions
LOG_FILE="${HOME}/.cloudwatch-dashboard-script.log"
touch "$LOG_FILE" && chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "$(date): Starting CloudWatch dashboard creation script"

# Security: Set strict error handling
set -uo pipefail
trap 'handle_error "Script failed at line $LINENO"' ERR

# Function to handle errors
handle_error() {
    local error_msg="${1:-Unknown error}"
    echo "ERROR: $error_msg" >&2
    echo "Resources created:"
    echo "- CloudWatch Dashboard: LambdaMetricsDashboard"
    echo ""
    echo "==========================================="
    echo "CLEANUP CONFIRMATION"
    echo "==========================================="
    echo "An error occurred. Proceeding with automatic cleanup..."

    echo "Cleaning up resources..."
    aws cloudwatch delete-dashboards --dashboard-names LambdaMetricsDashboard 2>/
dev/null || true
```

```
    # Clean up temporary files
    if [ -n "${TEMP_DIR:-}" ] && [ -d "$TEMP_DIR" ]; then
        rm -rf "$TEMP_DIR"
    fi
    rm -f dashboard-body.json

    echo "Cleanup complete."
    exit 1
}

# Security: Validate AWS CLI is installed
if ! command -v aws &> /dev/null; then
    handle_error "AWS CLI is not installed. Please install it and try again."
fi

# Check if AWS CLI is installed and configured
echo "Checking AWS CLI configuration..."
if ! aws sts get-caller-identity > /dev/null 2>&1; then
    handle_error "AWS CLI is not properly configured. Please configure it with
 'aws configure' and try again."
fi

# Get the current region securely
REGION=$(aws configure get region 2>/dev/null || echo "")
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $REGION"
fi
echo "Using region: $REGION"

# Validate region format
if ! [[ "$REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]{1}$ ]]; then
    handle_error "Invalid AWS region format: $REGION"
fi

# Check if there are any Lambda functions in the account
echo "Checking for Lambda functions..."
LAMBDA_FUNCTIONS=$(aws lambda list-functions --region "$REGION" --query
 "Functions[*].FunctionName" --output text 2>/dev/null || echo "")

if [ -z "$LAMBDA_FUNCTIONS" ]; then
    echo "No Lambda functions found in your account. Creating a simple test
 function..."
```

```
    # Create a temporary directory for Lambda function code with secure
 permissions
    TEMP_DIR=$(mktemp -d)
    chmod 700 "$TEMP_DIR"
    trap 'rm -rf "$TEMP_DIR"' EXIT

    # Create a simple Lambda function
    cat > "$TEMP_DIR/index.js" << 'EOF'
exports.handler = async (event) => {
    console.log('Event:', JSON.stringify(event, null, 2));
    return {
        statusCode: 200,
        body: JSON.stringify('Hello from Lambda!'),
    };
};
EOF

    # Zip the function code
    if ! cd "$TEMP_DIR"; then
        handle_error "Failed to change to temporary directory"
    fi

    if ! zip -q function.zip index.js; then
        handle_error "Failed to create zip file"
    fi

    # Create a role for the Lambda function with restricted trust policy
    ROLE_NAME="LambdaDashboardTestRole-$(date +%s)"
    TRUST_POLICY='{"Version":"2012-10-17",       "Statement":
[{"Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}'

    if ! ROLE_ARN=$(aws iam create-role \
        --role-name "$ROLE_NAME" \
        --assume-role-policy-document "$TRUST_POLICY" \
        --tags Key=project,Value=doc-smith Key=tutorial,Value=cloudwatch-
dynamicdash \
        --query "Role.Arn" \
        --output text 2>/dev/null); then
        handle_error "Failed to create IAM role for Lambda function"
    fi

    echo "Waiting for role to be available..."
```

```
    sleep 10

    # Attach basic Lambda execution policy
    if ! aws iam attach-role-policy \
        --role-name "$ROLE_NAME" \
        --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole"; then
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to attach policy to IAM role"
    fi

    # Create the Lambda function
    FUNCTION_NAME="DashboardTestFunction-$(date +%s)"
    if ! aws lambda create-function \
        --function-name "$FUNCTION_NAME" \
        --runtime nodejs18.x \
        --role "$ROLE_ARN" \
        --handler index.handler \
        --zip-file fileb://function.zip \
        --tags project=doc-smith,tutorial=cloudwatch-dynamicdash \
        --region "$REGION" > /dev/null 2>&1; then
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to create Lambda function"
    fi

    # Invoke the function to generate some metrics
    echo "Invoking Lambda function to generate metrics..."
    for i in {1..5}; do
        aws lambda invoke --function-name "$FUNCTION_NAME" --payload '{}' /dev/
null --region "$REGION" > /dev/null 2>&1 || true
        sleep 1
    done

    # Go back to original directory
    cd - > /dev/null

    # Set the function name for the dashboard
    DEFAULT_FUNCTION="$FUNCTION_NAME"
else
    # Use the first Lambda function as default
```

```
    DEFAULT_FUNCTION=$(echo "$LAMBDA_FUNCTIONS" | awk '{print $1}')
    echo "Found Lambda functions. Using $DEFAULT_FUNCTION as default."
    FUNCTION_NAME=""
    ROLE_NAME=""
fi

# Create a dashboard with Lambda metrics and a function name variable
echo "Creating CloudWatch dashboard with Lambda function name variable..."

# Create a JSON file for the dashboard body with secure permissions
DASHBOARD_JSON="dashboard-body-$$.json"
touch "$DASHBOARD_JSON" && chmod 600 "$DASHBOARD_JSON"

# Escape special characters in region and function name for JSON
REGION_ESCAPED=$(printf '%s\n' "$REGION" | sed 's:[\/&]:\\&:g')
FUNCTION_ESCAPED=$(printf '%s\n' "$DEFAULT_FUNCTION" | sed 's:[\/&]:\\&:g')

cat > "$DASHBOARD_JSON" << EOF
{
  "widgets": [
    {
      "type": "metric",
      "x": 0,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Invocations", "FunctionName", "\${FunctionName}" ],
          [ ".", "Errors", ".", "." ],
          [ ".", "Throttles", ".", "." ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Lambda Function Metrics for \${FunctionName}",
        "period": 300
      }
    },
    {
      "type": "metric",
      "x": 0,
      "y": 6,
      "width": 12,
```

```
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Duration", "FunctionName", "\${FunctionName}",
 { "stat": "Average" } ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Duration for \${FunctionName}",
        "period": 300
      }
    },
    {
      "type": "metric",
      "x": 12,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "ConcurrentExecutions", "FunctionName", "\
${FunctionName}" ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Concurrent Executions for \${FunctionName}",
        "period": 300
      }
    }
  ],
  "periodOverride": "auto",
  "variables": [
    {
      "type": "property",
      "id": "FunctionName",
      "property": "FunctionName",
      "label": "Lambda Function",
      "inputType": "select",
      "values": [
        {
          "value": "$FUNCTION_ESCAPED",
          "label": "$FUNCTION_ESCAPED"
```

```
        }
      ]
    }
  ]
}
EOF

# Validate JSON before sending
if ! jq empty "$DASHBOARD_JSON" 2>/dev/null; then
    handle_error "Invalid JSON generated for dashboard"
fi

# Create the dashboard using the JSON file
DASHBOARD_NAME="LambdaMetricsDashboard-$(date +%s)"
if ! DASHBOARD_RESULT=$(aws cloudwatch put-dashboard \
    --dashboard-name "$DASHBOARD_NAME" \
    --dashboard-body file://"$DASHBOARD_JSON" \
    --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to create CloudWatch dashboard."
fi

# Display any validation messages but continue
if echo "$DASHBOARD_RESULT" | grep -q "DashboardValidationMessages"; then
    echo "Dashboard created with validation messages:"
    echo "$DASHBOARD_RESULT"
    echo "These validation messages are warnings and the dashboard should still
 function."
else
    echo "Dashboard created successfully!"
fi

# Verify the dashboard was created
echo "Verifying dashboard creation..."
```

```
if ! DASHBOARD_INFO=$(aws cloudwatch get-dashboard --dashboard-name
 "$DASHBOARD_NAME" --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to verify dashboard creation."
fi

echo "Dashboard verification successful!"
echo "Dashboard details:"
echo "$DASHBOARD_INFO"

# List all dashboards to confirm
echo "Listing all dashboards:"
if ! DASHBOARDS=$(aws cloudwatch list-dashboards --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to list dashboards."
fi
echo "$DASHBOARDS"

# Show instructions for accessing the dashboard
echo ""
echo "Dashboard created successfully! To access it:"
echo "1. Open the CloudWatch console at https://console.aws.amazon.com/
cloudwatch/"
echo "2. In the navigation pane, choose Dashboards"
echo "3. Select $DASHBOARD_NAME"
```

```
echo "4. You should see a dropdown menu labeled 'Lambda Function' at the top of
 the dashboard"
echo "5. Use this dropdown to select different Lambda functions and see their
 metrics"
echo ""

# Create a list of resources for cleanup
RESOURCES=("- CloudWatch Dashboard: $DASHBOARD_NAME")
if [ -n "${FUNCTION_NAME:-}" ]; then
    RESOURCES+=("- Lambda Function: $FUNCTION_NAME")
    RESOURCES+=("- IAM Role: $ROLE_NAME")
fi

# Prompt for cleanup with automatic yes
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Resources created:"
for resource in "${RESOURCES[@]}"; do
    echo "$resource"
done
echo ""
echo "Proceeding with automatic cleanup..."

CLEANUP_CHOICE="y"

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."

    # Delete the dashboard
    if aws cloudwatch delete-dashboards --dashboard-names "$DASHBOARD_NAME" --
region "$REGION" 2>/dev/null; then
        echo "Dashboard deleted successfully."
    else
        echo "WARNING: Failed to delete dashboard. You may need to delete it
 manually."
    fi

    # If we created a Lambda function, delete it and its role
    if [ -n "${FUNCTION_NAME:-}" ]; then
        echo "Deleting Lambda function..."
        if aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null; then
            echo "Lambda function deleted successfully."
```

```
        else
            echo "WARNING: Failed to delete Lambda function. You may need to
 delete it manually."
        fi

        echo "Detaching role policy..."
        if aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null; then
            echo "Role policy detached successfully."
        else
            echo "WARNING: Failed to detach role policy. You may need to detach
 it manually."
        fi

        echo "Deleting IAM role..."
        if aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null; then
            echo "IAM role deleted successfully."
        else
            echo "WARNING: Failed to delete IAM role. You may need to delete it
 manually."
        fi
    fi

    # Clean up the JSON file
    rm -f "$DASHBOARD_JSON"

    echo "Cleanup complete."
fi

echo "Script completed successfully!"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteDashboards
  - DeleteFunction
  - DeleteRole

- DetachRolePolicy
- GetCallerIdentity
- GetDashboard
- Invoke
- ListDashboards
- ListFunctions
- PutDashboard

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Obtenez un jeton de session qui nécessite un jeton MFA avec AWS STS à l'aide d'un AWS Kit SDK

L'exemple de code suivant montre comment obtenir un jeton de session nécessitant un jeton MFA.

> **Warning**
>
> Afin d'éviter les risques de sécurité, n'employez pas les utilisateurs IAM pour l'authentification lorsque vous développez des logiciels spécialisés ou lorsque vous travaillez avec des données réelles. Préférez la fédération avec un fournisseur d'identité tel que AWS IAM Identity Center.

- Créez un rôle IAM qui accorde l'autorisation de répertorier les compartiments Amazon S3.
- Créez un utilisateur IAM autorisé à assumer le rôle uniquement lorsque des informations d'identification MFA sont fournies.
- Enregistrez un périphérique MFA pour l'utilisateur.
- Fournissez des informations d'identification MFA pour obtenir un jeton de session et utilisez des informations d'identification temporaires pour répertorier les compartiments S3.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez un utilisateur IAM, enregistrez un dispositif MFA et créez un rôle qui accorde l'autorisation de laisser l'utilisateur répertorier des compartiments S3 uniquement lorsque les informations d'identification MFA sont utilisées.

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual multi-factor authentication (MFA) device.
    Displays the QR code to seed the device.
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
    Creates an inline policy for the user that lets the user list Amazon S3
 buckets,
    but only when MFA credentials are used.

    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, MFA devices, and
                         policies in the account.
    :return: The newly created user, user key, and virtual MFA device.
    """
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")

    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
```

```
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")

    user_key = user.create_access_key_pair()
    print(f"Created access key pair for user.")

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    user.create_policy(
        PolicyName=unique_name("user-policy"),
        PolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
True}},
                    }
                ],
            }
        ),
    )
```

```
    print(
        f"Created an inline policy for {user.name} that lets the user list
 buckets, "
        f"but only when MFA credentials are present."
    )

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device
```

Obtenez des informations d'identification de session temporaires en transmettant un jeton MFA et utilisez les informations d'identification pour répertorier les compartiments S3 pour le compte.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
```

```
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

Détruisez les ressources créées pour la démo.

```
def teardown(user, virtual_mfa_device):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo MFA device.
    """
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
        key.delete()
        print("Deleted user's access key.")
    for mfa in user.mfa_devices.all():
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

Exécutez ce scénario à l'aide des fonctions définies précédemment.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(
```

```
        f"Welcome to the AWS Security Token Service assume role demo, "
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device = setup(iam_resource)
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
aws_secret_access_key=user_key.secret
        )
        try:
            print("Listing buckets without specifying MFA credentials.")
            list_buckets_with_session_token_with_mfa(None, None, sts_client)
        except ClientError as error:
            if error.response["Error"]["Code"] == "AccessDenied":
                print("Got expected AccessDenied error.")
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_with_session_token_with_mfa(
            virtual_mfa_device.serial_number, mfa_totp, sts_client
        )
    finally:
        teardown(user, virtual_mfa_device)
        print("Thanks for watching!")
```

- Pour plus de détails sur l'API, consultez GetSessionTokenle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Commencer à gérer la configuration

L'exemple de code suivant illustre comment :

- Créer un compartiment Amazon S3
- Créer une rubrique Amazon SNS

- Création d'un rôle IAM pour Config
- Configurer l'enregistreur de configuration Config
- Configurer le canal de diffusion Config
- Démarrez l'enregistreur de configuration
- Vérifiez la configuration de Config

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# AWS Config Setup Script (v2)
# This script sets up AWS Config with the AWS CLI

# Error handling
set -e
LOGFILE="aws-config-setup-v2.log"
touch $LOGFILE
exec > >(tee -a $LOGFILE)
exec 2>&1

# Function to handle errors
handle_error() {
    echo "ERROR: An error occurred at line $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set trap for error handling
trap 'handle_error $LINENO' ERR

# Function to generate random identifier
```

```
generate_random_id() {
    echo $(openssl rand -hex 6)
}

# Function to check if command was successful
check_command() {
    if echo "$1" | grep -i "error" > /dev/null; then
        echo "ERROR: $1"
        return 1
    fi
    return 0
}

# Function to clean up resources
cleanup_resources() {
    if [ -n "$CONFIG_RECORDER_NAME" ]; then
        echo "Stopping configuration recorder..."
        aws configservice stop-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    # Check if we created a new delivery channel before trying to delete it
    if [ -n "$DELIVERY_CHANNEL_NAME" ] && [ "$CREATED_NEW_DELIVERY_CHANNEL" =
 "true" ]; then
        echo "Deleting delivery channel..."
        aws configservice delete-delivery-channel --delivery-channel-name
 "$DELIVERY_CHANNEL_NAME" 2>/dev/null || true
    fi

    if [ -n "$CONFIG_RECORDER_NAME" ] && [ "$CREATED_NEW_CONFIG_RECORDER" =
 "true" ]; then
        echo "Deleting configuration recorder..."
        aws configservice delete-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    if [ -n "$ROLE_NAME" ]; then
        if [ -n "$POLICY_NAME" ]; then
            echo "Detaching custom policy from role..."
            aws iam delete-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" 2>/dev/null || true
        fi

        if [ -n "$MANAGED_POLICY_ARN" ]; then
```

```
            echo "Detaching managed policy from role..."
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN" 2>/dev/null || true
        fi

        echo "Deleting IAM role..."
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi

    if [ -n "$SNS_TOPIC_ARN" ]; then
        echo "Deleting SNS topic..."
        aws sns delete-topic --topic-arn "$SNS_TOPIC_ARN" 2>/dev/null || true
    fi

    if [ -n "$S3_BUCKET_NAME" ]; then
        echo "Emptying S3 bucket..."
        aws s3 rm "s3://$S3_BUCKET_NAME" --recursive 2>/dev/null || true

        echo "Deleting S3 bucket..."
        if [ "$BUCKET_IS_SHARED" = "false" ]; then
            aws s3api delete-bucket --bucket "$S3_BUCKET_NAME" 2>/dev/null ||
 true
        fi
    fi
}

# Function to display created resources
display_resources() {
    echo ""
    echo "========================================="
    echo "CREATED RESOURCES"
    echo "========================================="
    echo "S3 Bucket: $S3_BUCKET_NAME"
    echo "SNS Topic ARN: $SNS_TOPIC_ARN"
    echo "IAM Role: $ROLE_NAME"
    if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (newly created)"
    else
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (existing)"
    fi
    if [ "$CREATED_NEW_DELIVERY_CHANNEL" = "true" ]; then
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (newly created)"
    else
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (existing)"
```

```
    fi
    echo "========================================="
}

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
if [ -z "$ACCOUNT_ID" ]; then
    echo "ERROR: Failed to get AWS account ID"
    exit 1
fi
echo "AWS Account ID: $ACCOUNT_ID"

# Generate random identifier for resources
RANDOM_ID=$(generate_random_id)
echo "Generated random identifier: $RANDOM_ID"

# Step 1: Create an S3 bucket
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="configservice-${RANDOM_ID}"
    echo "Creating S3 bucket: $S3_BUCKET_NAME"
fi

# Get the current region
AWS_REGION=$(aws configure get region)
if [ -z "$AWS_REGION" ]; then
    AWS_REGION="us-east-1"  # Default to us-east-1 if no region is configured
fi
echo "Using AWS Region: $AWS_REGION"

# Create bucket with appropriate command based on region
if [ "$BUCKET_IS_SHARED" = "false" ]; then
    if [ "$AWS_REGION" = "us-east-1" ]; then
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME")
```

```
    else
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" --
create-bucket-configuration LocationConstraint="$AWS_REGION")
    fi
    check_command "$BUCKET_RESULT"
    echo "S3 bucket created: $S3_BUCKET_NAME"

    aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-config-gs}]'
    echo "Tags applied to S3 bucket"
else
    echo "Using shared bucket: $S3_BUCKET_NAME (skipping creation)"
fi

# Block public access for the bucket
aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
echo "Public access blocked for bucket"

# Step 2: Create an SNS topic
TOPIC_NAME="config-topic-${RANDOM_ID}"
echo "Creating SNS topic: $TOPIC_NAME"
SNS_RESULT=$(aws sns create-topic --name "$TOPIC_NAME" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs)
check_command "$SNS_RESULT"
SNS_TOPIC_ARN=$(echo "$SNS_RESULT" | grep -o 'arn:aws:sns:[^"]*')
echo "SNS topic created: $SNS_TOPIC_ARN"

# Step 3: Create an IAM role for AWS Config
ROLE_NAME="config-role-${RANDOM_ID}"
POLICY_NAME="config-delivery-permissions"
MANAGED_POLICY_ARN="arn:aws:iam::aws:policy/service-role/AWS_ConfigRole"

echo "Creating trust policy document..."
cat > config-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "config.amazonaws.com"
```

```
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role --role-name "$ROLE_NAME" --assume-role-policy-
document file://config-trust-policy.json)
check_command "$ROLE_RESULT"
ROLE_ARN=$(echo "$ROLE_RESULT" | grep -o 'arn:aws:iam::[^"]*' | head -1)
echo "IAM role created: $ROLE_ARN"

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-config-gs
echo "Tags applied to IAM role"

echo "Attaching AWS managed policy to role..."
ATTACH_RESULT=$(aws iam attach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN")
check_command "$ATTACH_RESULT"
echo "AWS managed policy attached"

echo "Creating custom policy document for S3 and SNS access..."
cat > config-delivery-permissions.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}/AWSLogs/${ACCOUNT_ID}/*",
      "Condition": {
        "StringLike": {
          "s3:x-amz-acl": "bucket-owner-full-control"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
```

```
        "s3:GetBucketAcl"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}"
    },
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": "${SNS_TOPIC_ARN}"
    }
  ]
}
EOF

echo "Attaching custom policy to role..."
POLICY_RESULT=$(aws iam put-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" --policy-document file://config-delivery-permissions.json)
check_command "$POLICY_RESULT"
echo "Custom policy attached"

# Wait for IAM role to propagate
echo "Waiting for IAM role to propagate (15 seconds)..."
sleep 15

# Step 4: Check if configuration recorder already exists
CONFIG_RECORDER_NAME="default"
CREATED_NEW_CONFIG_RECORDER="false"

echo "Checking for existing configuration recorder..."
EXISTING_RECORDERS=$(aws configservice describe-configuration-recorders 2>/dev/
null || echo "")
if echo "$EXISTING_RECORDERS" | grep -q "name"; then
    echo "Configuration recorder already exists. Will update it."
    # Get the name of the existing recorder
    CONFIG_RECORDER_NAME=$(echo "$EXISTING_RECORDERS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
    echo "Using existing configuration recorder: $CONFIG_RECORDER_NAME"
else
    echo "No existing configuration recorder found. Will create a new one."
    CREATED_NEW_CONFIG_RECORDER="true"
fi

echo "Creating configuration recorder configuration..."
```

```
cat > configurationRecorder.json << EOF
{
  "name": "${CONFIG_RECORDER_NAME}",
  "roleARN": "${ROLE_ARN}",
  "recordingMode": {
    "recordingFrequency": "CONTINUOUS"
  }
}
EOF

echo "Creating recording group configuration..."
cat > recordingGroup.json << EOF
{
  "allSupported": true,
  "includeGlobalResourceTypes": true
}
EOF

echo "Setting up configuration recorder..."
RECORDER_RESULT=$(aws configservice put-configuration-recorder --configuration-
recorder file://configurationRecorder.json --recording-group file://
recordingGroup.json)
check_command "$RECORDER_RESULT"
echo "Configuration recorder set up"

if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:config-recorder/${CONFIG_RECORDER_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to configuration recorder"
fi

# Step 5: Check if delivery channel already exists
DELIVERY_CHANNEL_NAME="default"
CREATED_NEW_DELIVERY_CHANNEL="false"

echo "Checking for existing delivery channel..."
EXISTING_CHANNELS=$(aws configservice describe-delivery-channels 2>/dev/null ||
 echo "")
if echo "$EXISTING_CHANNELS" | grep -q "name"; then
    echo "Delivery channel already exists."
    # Get the name of the existing channel
    DELIVERY_CHANNEL_NAME=$(echo "$EXISTING_CHANNELS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
```

```
    echo "Using existing delivery channel: $DELIVERY_CHANNEL_NAME"

    # Update the existing delivery channel
    echo "Creating delivery channel configuration for update..."
    cat > deliveryChannel.json << EOF
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Updating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel updated"
else
    echo "No existing delivery channel found. Will create a new one."
    CREATED_NEW_DELIVERY_CHANNEL="true"

    echo "Creating delivery channel configuration..."
    cat > deliveryChannel.json << EOF
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Creating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel created"
```

```
    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:delivery-channel/${DELIVERY_CHANNEL_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to delivery channel"
fi

# Step 6: Start the configuration recorder
echo "Checking configuration recorder status..."
RECORDER_STATUS=$(aws configservice describe-configuration-recorder-status 2>/
dev/null || echo "")
if echo "$RECORDER_STATUS" | grep -q '"recording": true'; then
    echo "Configuration recorder is already running."
else
    echo "Starting configuration recorder..."
    START_RESULT=$(aws configservice start-configuration-recorder --
configuration-recorder-name "$CONFIG_RECORDER_NAME")
    check_command "$START_RESULT"
    echo "Configuration recorder started"
fi

# Step 7: Verify the AWS Config setup
echo "Verifying delivery channel..."
VERIFY_CHANNEL=$(aws configservice describe-delivery-channels)
check_command "$VERIFY_CHANNEL"
echo "$VERIFY_CHANNEL"

echo "Verifying configuration recorder..."
VERIFY_RECORDER=$(aws configservice describe-configuration-recorders)
check_command "$VERIFY_RECORDER"
echo "$VERIFY_RECORDER"

echo "Verifying configuration recorder status..."
VERIFY_STATUS=$(aws configservice describe-configuration-recorder-status)
check_command "$VERIFY_STATUS"
echo "$VERIFY_STATUS"

# Display created resources
display_resources

# Ask if user wants to clean up resources
echo ""
echo "============================================"
echo "CLEANUP CONFIRMATION"
echo "============================================"
```

```
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE='y'

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..."
    cleanup_resources
    echo "Cleanup completed."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
fi

echo "Script completed successfully!"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreateBucket
  - CreateRole
  - CreateTopic
  - DeleteBucket
  - DeleteConfigurationRecorder
  - DeleteDeliveryChannel
  - DeleteRole
  - DeleteRolePolicy
  - DeleteTopic
  - DescribeConfigurationRecorderStatus
  - DescribeConfigurationRecorders
  - DescribeDeliveryChannels
  - DetachRolePolicy
  - GetCallerIdentity
  - PutConfigurationRecorder
  - PutDeliveryChannel
  - PutPublicAccessBlock
  - PutRolePolicy

- Rm
- StartConfigurationRecorder
- StopConfigurationRecorder

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Commencer à utiliser les registres de conteneurs

L'exemple de code suivant illustre comment :

- Créer une image Docker
- Création d'un référentiel Amazon ECR
- Supprimer des ressources

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Amazon ECR Getting Started Script
# This script demonstrates the lifecycle of a Docker image in Amazon ECR

# Set up logging
LOG_FILE="ecr-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "================================================="
echo "Amazon ECR Getting Started Tutorial"
echo "================================================="
echo "This script will:"
```

```
echo "1. Create a Docker image"
echo "2. Create an Amazon ECR repository"
echo "3. Authenticate to Amazon ECR"
echo "4. Push the image to Amazon ECR"
echo "5. Pull the image from Amazon ECR"
echo "6. Clean up resources (optional)"
echo "================================================="

# Check prerequisites
echo "Checking prerequisites..."

# Check if AWS CLI is installed
if ! command -v aws &> /dev/null; then
    echo "ERROR: AWS CLI is not installed. Please install it before running this
 script."
    echo "Visit https://docs.aws.amazon.com/cli/latest/userguide/install-
cliv2.html for installation instructions."
    exit 1
fi

# Check if AWS CLI is configured
if ! aws sts get-caller-identity &> /dev/null; then
    echo "ERROR: AWS CLI is not configured properly. Please run 'aws configure'
 to set up your credentials."
    exit 1
fi

# Check if Docker is installed
if ! command -v docker &> /dev/null; then
    echo "ERROR: Docker is not installed. Please install Docker before running
 this script."
    echo "Visit https://docs.docker.com/get-docker/ for installation
 instructions."
    exit 1
fi

# Check if Docker daemon is running
if ! docker info &> /dev/null; then
    echo "ERROR: Docker daemon is not running. Please start Docker and try
 again."
    exit 1
fi

echo "All prerequisites met."
```

```
# Initialize variables
REPO_URI=""
TIMEOUT_CMD="timeout 300"  # 5-minute timeout for long-running commands

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Check the log file for details: $LOG_FILE"

    echo "=================================================="
    echo "Resources created:"
    echo "- Docker image: hello-world (local)"
    if [ -n "$REPO_URI" ]; then
        echo "- ECR Repository: hello-repository"
        echo "- ECR Image: $REPO_URI:latest"
    fi
    echo "=================================================="

    echo "Attempting to clean up resources..."
    cleanup
    exit 1
}

# Function to clean up resources
cleanup() {
    echo "=================================================="
    echo "Cleaning up resources..."

    # Delete the image from ECR if it exists
    if [ -n "$REPO_URI" ]; then
        echo "Deleting image from ECR repository..."
        aws ecr batch-delete-image --repository-name hello-repository --image-ids
 imageTag=latest || echo "Failed to delete image, it may not exist or may have
 already been deleted."
    fi

    # Delete the ECR repository if it exists
    if [ -n "$REPO_URI" ]; then
        echo "Deleting ECR repository..."
        aws ecr delete-repository --repository-name hello-repository --force ||
 echo "Failed to delete repository, it may not exist or may have already been
 deleted."
    fi
```

```
    # Remove local Docker image
    echo "Removing local Docker image..."
    docker rmi hello-world:latest 2>/dev/null || echo "Failed to remove local
 image, it may not exist or may have already been deleted."
    if [ -n "$REPO_URI" ]; then
        docker rmi "$REPO_URI:latest" 2>/dev/null || echo "Failed to remove
 tagged image, it may not exist or may have already been deleted."
    fi

    echo "Cleanup completed."
    echo "================================================="
}

# Step 1: Create a Docker image
echo "Step 1: Creating a Docker image"

# Create Dockerfile
echo "Creating Dockerfile..."
cat > Dockerfile << 'EOF'
FROM public.ecr.aws/amazonlinux/amazonlinux:latest

# Install dependencies
RUN yum update -y && \
 yum install -y httpd

# Install apache and write hello world message
RUN echo 'Hello World!' > /var/www/html/index.html

# Configure apache
RUN echo 'mkdir -p /var/run/httpd' >> /root/run_apache.sh && \
 echo 'mkdir -p /var/lock/httpd' >> /root/run_apache.sh && \
 echo '/usr/sbin/httpd -D FOREGROUND' >> /root/run_apache.sh && \
 chmod 755 /root/run_apache.sh

EXPOSE 80

CMD /root/run_apache.sh
EOF

# Build Docker image
echo "Building Docker image..."
$TIMEOUT_CMD docker build -t hello-world . || handle_error "Failed to build
 Docker image or operation timed out after 5 minutes"
```

```
# Verify image was created
echo "Verifying Docker image..."
docker images --filter reference=hello-world || handle_error "Failed to list
 Docker images"

echo "Docker image created successfully."

# Step 2: Create an Amazon ECR repository
echo "Step 2: Creating an Amazon ECR repository"

# Get AWS account ID
echo "Getting AWS account ID..."
AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ -z "$AWS_ACCOUNT_ID" || "$AWS_ACCOUNT_ID" == *"error"* ]]; then
    handle_error "Failed to get AWS account ID. Make sure your AWS credentials
 are configured correctly."
fi
echo "AWS Account ID: $AWS_ACCOUNT_ID"

# Get current region
AWS_REGION=$(aws configure get region)
if [[ -z "$AWS_REGION" ]]; then
    AWS_REGION="us-east-1"  # Default to us-east-1 if no region is configured
    echo "No AWS region configured, defaulting to $AWS_REGION"
else
    echo "Using AWS region: $AWS_REGION"
fi

# Create ECR repository
echo "Creating ECR repository..."
REPO_RESULT=$(aws ecr create-repository --repository-name hello-repository --
tags key=project,value=doc-smith key=tutorial,value=amazon-elastic-container-
registry-gs)
if [[ -z "$REPO_RESULT" || "$REPO_RESULT" == *"error"* ]]; then
    handle_error "Failed to create ECR repository"
fi

# Extract repository URI
REPO_URI="$AWS_ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/hello-repository"
echo "Repository URI: $REPO_URI"

# Step 3: Authenticate to Amazon ECR
echo "Step 3: Authenticating to Amazon ECR"
```

```
echo "Getting ECR login password..."
aws ecr get-login-password --region "$AWS_REGION" | docker login --username
 AWS --password-stdin "$AWS_ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com" ||
 handle_error "Failed to authenticate to ECR"

echo "Successfully authenticated to ECR."

# Step 4: Push the image to Amazon ECR
echo "Step 4: Pushing the image to Amazon ECR"

# Tag the image
echo "Tagging Docker image..."
docker tag hello-world:latest "$REPO_URI:latest" || handle_error "Failed to tag
 Docker image"

# Push the image with timeout
echo "Pushing image to ECR..."
$TIMEOUT_CMD docker push "$REPO_URI:latest" || handle_error "Failed to push image
 to ECR or operation timed out after 5 minutes"

echo "Successfully pushed image to ECR."

# Step 5: Pull the image from Amazon ECR
echo "Step 5: Pulling the image from Amazon ECR"

# Remove local image to ensure we're pulling from ECR
echo "Removing local tagged image..."
docker rmi "$REPO_URI:latest" || echo "Warning: Failed to remove local tagged
 image"

# Pull the image with timeout
echo "Pulling image from ECR..."
$TIMEOUT_CMD docker pull "$REPO_URI:latest" || handle_error "Failed to pull image
 from ECR or operation timed out after 5 minutes"

echo "Successfully pulled image from ECR."

# List resources created
echo "================================================="
echo "Resources created:"
echo "- Docker image: hello-world (local)"
echo "- ECR Repository: hello-repository"
echo "- ECR Image: $REPO_URI:latest"
```

```
echo "================================================="

# Ask user if they want to clean up resources
echo ""
echo "=========================================="
echo "CLEANUP CONFIRMATION"
echo "=========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    # Step 6: Delete the image from ECR
    echo "Step 6: Deleting the image from ECR"

    DELETE_IMAGE_RESULT=$(aws ecr batch-delete-image --repository-name hello-
repository --image-ids imageTag=latest)
    if [[ -z "$DELETE_IMAGE_RESULT" || "$DELETE_IMAGE_RESULT" == *"error"* ]];
 then
        echo "Warning: Failed to delete image from ECR"
    else
        echo "Successfully deleted image from ECR."
    fi

    # Step 7: Delete the ECR repository
    echo "Step 7: Deleting the ECR repository"

    DELETE_REPO_RESULT=$(aws ecr delete-repository --repository-name hello-
repository --force)
    if [[ -z "$DELETE_REPO_RESULT" || "$DELETE_REPO_RESULT" == *"error"* ]]; then
        echo "Warning: Failed to delete ECR repository"
    else
        echo "Successfully deleted ECR repository."
    fi

    # Remove local Docker images
    echo "Removing local Docker images..."
    docker rmi hello-world:latest 2>/dev/null || echo "Warning: Failed to remove
 local image"

    echo "All resources have been cleaned up."
else
    echo "Resources were not cleaned up. You can manually clean up later with:"
    echo "aws ecr batch-delete-image --repository-name hello-repository --image-
ids imageTag=latest"
```

```
    echo "aws ecr delete-repository --repository-name hello-repository --force"
    echo "docker rmi hello-world:latest"
    echo "docker rmi $REPO_URI:latest"
fi

echo "================================================="
echo "Tutorial completed!"
echo "Log file: $LOG_FILE"
echo "================================================="
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - BatchDeleteImage
  - CreateRepository
  - DeleteRepository
  - GetCallerIdentity
  - GetLoginPassword

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Débuter avec la messagerie de l'Internet des objets

L'exemple de code suivant illustre comment :

- Créez des ressources IoT
- Configurer votre appareil
- Exécuter les exemples d'applications
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# AWS IoT Core Getting Started Script
# This script creates AWS IoT resources, configures a device, and runs a sample
 application

# Set up logging
LOG_FILE="iot-core-setup.log"
echo "Starting AWS IoT Core setup at $(date)" > $LOG_FILE

# Function to log commands and their outputs
log_cmd() {
    echo "$(date): Running command: $1" >> $LOG_FILE
    eval "$1" 2>&1 | tee -a $LOG_FILE
    return ${PIPESTATUS[0]}
}

# Function to check for errors
check_error() {
    if [ $1 -ne 0 ]; then
        echo "ERROR: Command failed with exit code $1" | tee -a $LOG_FILE
        echo "Please check the log file $LOG_FILE for details" | tee -a $LOG_FILE
        cleanup_on_error
        exit $1
    fi
}

# Function to cleanup resources on error
cleanup_on_error() {
    echo "Error encountered. Attempting to clean up resources..." | tee -a
 $LOG_FILE
    echo "Resources created:" | tee -a $LOG_FILE
```

```
    if [ ! -z "$CERTIFICATE_ARN" ]; then
        echo "Certificate ARN: $CERTIFICATE_ARN" | tee -a $LOG_FILE
        if [ ! -z "$POLICY_NAME" ]; then
            log_cmd "aws iot detach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN"
        fi
        if [ ! -z "$THING_NAME" ]; then
            log_cmd "aws iot detach-thing-principal --thing-name $THING_NAME --
principal $CERTIFICATE_ARN"
        fi
        if [ ! -z "$CERTIFICATE_ID" ]; then
            log_cmd "aws iot update-certificate --certificate-id $CERTIFICATE_ID
 --new-status INACTIVE"
            log_cmd "aws iot delete-certificate --certificate-id $CERTIFICATE_ID"
        fi
    fi
    if [ ! -z "$THING_NAME" ]; then
        echo "Thing Name: $THING_NAME" | tee -a $LOG_FILE
        log_cmd "aws iot delete-thing --thing-name $THING_NAME"
    fi
    if [ ! -z "$POLICY_NAME" ]; then
        echo "Policy Name: $POLICY_NAME" | tee -a $LOG_FILE
        log_cmd "aws iot delete-policy --policy-name $POLICY_NAME"
    fi
    if [ ! -z "$SHARED_POLICY_NAME" ]; then
        echo "Shared Policy Name: $SHARED_POLICY_NAME" | tee -a $LOG_FILE
        log_cmd "aws iot delete-policy --policy-name $SHARED_POLICY_NAME"
    fi
}

# Generate unique identifiers
RANDOM_SUFFIX=$(openssl rand -hex 4)
THING_NAME="MyIoTThing-${RANDOM_SUFFIX}"
POLICY_NAME="MyIoTPolicy-${RANDOM_SUFFIX}"
SHARED_POLICY_NAME="SharedSubPolicy-${RANDOM_SUFFIX}"
CERTS_DIR="$HOME/certs"

echo "================================================" | tee -a $LOG_FILE
echo "AWS IoT Core Getting Started" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "This script will:" | tee -a $LOG_FILE
echo "1. Create AWS IoT resources (policy, thing, certificate)" | tee -a
 $LOG_FILE
echo "2. Configure your device" | tee -a $LOG_FILE
```

```
echo "3. Set up for running the sample application" | tee -a $LOG_FILE
echo "" | tee -a $LOG_FILE
echo "Thing Name: $THING_NAME" | tee -a $LOG_FILE
echo "Policy Name: $POLICY_NAME" | tee -a $LOG_FILE
echo "Certificates Directory: $CERTS_DIR" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "" | tee -a $LOG_FILE

# Get AWS account ID
echo "Getting AWS account ID..." | tee -a $LOG_FILE
ACCOUNT_ID=$(log_cmd "aws sts get-caller-identity --query Account --output text")
check_error $?

# Get AWS region
echo "Getting AWS region..." | tee -a $LOG_FILE
REGION=$(log_cmd "aws configure get region")
check_error $?
if [ -z "$REGION" ]; then
    echo "AWS region not configured. Please run 'aws configure' to set your
 region." | tee -a $LOG_FILE
    exit 1
fi

echo "Using AWS Account ID: $ACCOUNT_ID and Region: $REGION" | tee -a $LOG_FILE

# Step 1: Create AWS IoT Resources
echo "" | tee -a $LOG_FILE
echo "Step 1: Creating AWS IoT Resources..." | tee -a $LOG_FILE

# Create IoT policy
echo "Creating IoT policy document..." | tee -a $LOG_FILE
cat > iot-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iot:Connect"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:client/test-*"
      ]
    },
```

```
    {
      "Effect": "Allow",
      "Action": [
        "iot:Publish",
        "iot:Receive"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topic/test/topic"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Subscribe"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topicfilter/test/topic"
      ]
    }
  ]
}
EOF

echo "Creating IoT policy: $POLICY_NAME..." | tee -a $LOG_FILE
log_cmd "aws iot create-policy --policy-name $POLICY_NAME --policy-
document file://iot-policy.json --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-core-gs"
check_error $?

# Create IoT thing
echo "Creating IoT thing: $THING_NAME..." | tee -a $LOG_FILE
log_cmd "aws iot create-thing --thing-name $THING_NAME"
check_error $?

THING_ARN=$(aws iot describe-thing --thing-name "$THING_NAME" --query 'thingArn'
 --output text)
aws iot tag-resource --resource-arn "$THING_ARN" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-iot-core-gs

# Create directory for certificates
echo "Creating certificates directory..." | tee -a $LOG_FILE
log_cmd "mkdir -p $CERTS_DIR"
check_error $?
```

```
# Create keys and certificate
echo "Creating keys and certificate..." | tee -a $LOG_FILE
CERT_OUTPUT=$(log_cmd "aws iot create-keys-and-certificate --set-as-active
 --certificate-pem-outfile $CERTS_DIR/device.pem.crt --public-key-outfile
 $CERTS_DIR/public.pem.key --private-key-outfile $CERTS_DIR/private.pem.key")
check_error $?

# Extract certificate ARN and ID
CERTIFICATE_ARN=$(echo "$CERT_OUTPUT" | grep "certificateArn" | cut -d'"' -f4)
CERTIFICATE_ID=$(echo "$CERTIFICATE_ARN" | cut -d/ -f2)

if [ -z "$CERTIFICATE_ARN" ] || [ -z "$CERTIFICATE_ID" ]; then
    echo "Failed to extract certificate ARN or ID" | tee -a $LOG_FILE
    cleanup_on_error
    exit 1
fi

echo "Certificate ARN: $CERTIFICATE_ARN" | tee -a $LOG_FILE
echo "Certificate ID: $CERTIFICATE_ID" | tee -a $LOG_FILE

# Attach policy to certificate
echo "Attaching policy to certificate..." | tee -a $LOG_FILE
log_cmd "aws iot attach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN"
check_error $?

# Attach certificate to thing
echo "Attaching certificate to thing..." | tee -a $LOG_FILE
log_cmd "aws iot attach-thing-principal --thing-name $THING_NAME --principal
 $CERTIFICATE_ARN"
check_error $?

# Download Amazon Root CA certificate
echo "Downloading Amazon Root CA certificate..." | tee -a $LOG_FILE
log_cmd "curl -s -o $CERTS_DIR/Amazon-root-CA-1.pem https://www.amazontrust.com/
repository/AmazonRootCA1.pem"
check_error $?

# Step 2: Configure Your Device
echo "" | tee -a $LOG_FILE
echo "Step 2: Configuring Your Device..." | tee -a $LOG_FILE

# Check if Git is installed
echo "Checking if Git is installed..." | tee -a $LOG_FILE
```

```
if ! command -v git &> /dev/null; then
    echo "Git is not installed. Please install Git and run this script again." |
 tee -a $LOG_FILE
    cleanup_on_error
    exit 1
fi

# Check if Python is installed
echo "Checking if Python is installed..." | tee -a $LOG_FILE
if ! command -v python3 &> /dev/null; then
    echo "Python 3 is not installed. Please install Python 3 and run this script
 again." | tee -a $LOG_FILE
    cleanup_on_error
    exit 1
fi

# Install AWS IoT Device SDK for Python
echo "Installing AWS IoT Device SDK for Python..." | tee -a $LOG_FILE
log_cmd "python3 -m pip install awsiotsdk"
check_error $?

# Clone the AWS IoT Device SDK for Python repository
echo "Cloning AWS IoT Device SDK for Python repository..." | tee -a $LOG_FILE
if [ ! -d "$HOME/aws-iot-device-sdk-python-v2" ]; then
    log_cmd "cd $HOME && git clone https://github.com/aws/aws-iot-device-sdk-
python-v2.git"
    check_error $?
else
    echo "AWS IoT Device SDK for Python repository already exists." | tee -a
 $LOG_FILE
fi

# Step 3: Get AWS IoT Endpoint
echo "" | tee -a $LOG_FILE
echo "Step 3: Getting AWS IoT Endpoint..." | tee -a $LOG_FILE

IOT_ENDPOINT=$(log_cmd "aws iot describe-endpoint --endpoint-type iot:Data-ATS --
query endpointAddress --output text")
check_error $?

echo "AWS IoT Endpoint: $IOT_ENDPOINT" | tee -a $LOG_FILE

# Create a shared subscription policy (optional)
echo "" | tee -a $LOG_FILE
```

```
echo "Creating shared subscription policy (optional)..." | tee -a $LOG_FILE

cat > shared-sub-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iot:Connect"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:client/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Publish",
        "iot:Receive"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topic/test/topic"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Subscribe"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topicfilter/test/topic",
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topicfilter/\$share/*/test/topic"
      ]
    }
  ]
}
EOF

log_cmd "aws iot create-policy --policy-name $SHARED_POLICY_NAME --policy-
document file://shared-sub-policy.json --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-core-gs"
check_error $?
```

```
log_cmd "aws iot attach-policy --policy-name $SHARED_POLICY_NAME --target
 $CERTIFICATE_ARN"
check_error $?

# Summary of created resources
echo "" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "Setup Complete! Resources Created:" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "Thing Name: $THING_NAME" | tee -a $LOG_FILE
echo "Policy Name: $POLICY_NAME" | tee -a $LOG_FILE
echo "Shared Subscription Policy Name: $SHARED_POLICY_NAME" | tee -a $LOG_FILE
echo "Certificate ID: $CERTIFICATE_ID" | tee -a $LOG_FILE
echo "Certificate ARN: $CERTIFICATE_ARN" | tee -a $LOG_FILE
echo "Certificate Files Location: $CERTS_DIR" | tee -a $LOG_FILE
echo "AWS IoT Endpoint: $IOT_ENDPOINT" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE

# Instructions for running the sample application
echo "" | tee -a $LOG_FILE
echo "To run the sample application, execute:" | tee -a $LOG_FILE
echo "cd $HOME/aws-iot-device-sdk-python-v2/samples" | tee -a $LOG_FILE
echo "python3 pubsub.py \\" | tee -a $LOG_FILE
echo "  --endpoint $IOT_ENDPOINT \\" | tee -a $LOG_FILE
echo "  --ca_file $CERTS_DIR/Amazon-root-CA-1.pem \\" | tee -a $LOG_FILE
echo "  --cert $CERTS_DIR/device.pem.crt \\" | tee -a $LOG_FILE
echo "  --key $CERTS_DIR/private.pem.key" | tee -a $LOG_FILE
echo "" | tee -a $LOG_FILE
echo "To run the shared subscription example, execute:" | tee -a $LOG_FILE
echo "cd $HOME/aws-iot-device-sdk-python-v2/samples" | tee -a $LOG_FILE
echo "python3 mqtt5_shared_subscription.py \\" | tee -a $LOG_FILE
echo "  --endpoint $IOT_ENDPOINT \\" | tee -a $LOG_FILE
echo "  --ca_file $CERTS_DIR/Amazon-root-CA-1.pem \\" | tee -a $LOG_FILE
echo "  --cert $CERTS_DIR/device.pem.crt \\" | tee -a $LOG_FILE
echo "  --key $CERTS_DIR/private.pem.key \\" | tee -a $LOG_FILE
echo "  --group_identifier consumer" | tee -a $LOG_FILE

# Ask if user wants to clean up resources
echo "" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "CLEANUP CONFIRMATION" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "Do you want to clean up all created resources? (y/n): " | tee -a $LOG_FILE
read -r CLEANUP_CHOICE
```

```
if [[ $CLEANUP_CHOICE =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..." | tee -a $LOG_FILE

    # Detach policies from certificate
    echo "Detaching policies from certificate..." | tee -a $LOG_FILE
    log_cmd "aws iot detach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN"
    log_cmd "aws iot detach-policy --policy-name $SHARED_POLICY_NAME --target
 $CERTIFICATE_ARN"

    # Detach certificate from thing
    echo "Detaching certificate from thing..." | tee -a $LOG_FILE
    log_cmd "aws iot detach-thing-principal --thing-name $THING_NAME --principal
 $CERTIFICATE_ARN"

    # Update certificate status to INACTIVE
    echo "Setting certificate to inactive..." | tee -a $LOG_FILE
    log_cmd "aws iot update-certificate --certificate-id $CERTIFICATE_ID --new-
status INACTIVE"

    # Delete certificate
    echo "Deleting certificate..." | tee -a $LOG_FILE
    log_cmd "aws iot delete-certificate --certificate-id $CERTIFICATE_ID"

    # Delete thing
    echo "Deleting thing..." | tee -a $LOG_FILE
    log_cmd "aws iot delete-thing --thing-name $THING_NAME"

    # Delete policies
    echo "Deleting policies..." | tee -a $LOG_FILE
    log_cmd "aws iot delete-policy --policy-name $POLICY_NAME"
    log_cmd "aws iot delete-policy --policy-name $SHARED_POLICY_NAME"

    echo "Cleanup complete!" | tee -a $LOG_FILE
else
    echo "Resources were not cleaned up. You can manually clean them up later." |
 tee -a $LOG_FILE
    echo "To clean up resources, run the following commands:" | tee -a $LOG_FILE
    echo "aws iot detach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN" | tee -a $LOG_FILE
    echo "aws iot detach-policy --policy-name $SHARED_POLICY_NAME --target
 $CERTIFICATE_ARN" | tee -a $LOG_FILE
```

```
    echo "aws iot detach-thing-principal --thing-name $THING_NAME --principal
 $CERTIFICATE_ARN" | tee -a $LOG_FILE
    echo "aws iot update-certificate --certificate-id $CERTIFICATE_ID --new-
status INACTIVE" | tee -a $LOG_FILE
    echo "aws iot delete-certificate --certificate-id $CERTIFICATE_ID" | tee -a
 $LOG_FILE
    echo "aws iot delete-thing --thing-name $THING_NAME" | tee -a $LOG_FILE
    echo "aws iot delete-policy --policy-name $POLICY_NAME" | tee -a $LOG_FILE
    echo "aws iot delete-policy --policy-name $SHARED_POLICY_NAME" | tee -a
 $LOG_FILE
fi

echo "" | tee -a $LOG_FILE
echo "Script execution completed. See $LOG_FILE for details." | tee -a $LOG_FILE
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachPolicy
  - AttachThingPrincipal
  - CreateKeysAndCertificate
  - CreatePolicy
  - CreateThing
  - DeleteCertificate
  - DeletePolicy
  - DeleteThing
  - DescribeEndpoint
  - DetachPolicy
  - DetachThingPrincipal
  - GetCallerIdentity
  - UpdateCertificate

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Commencer à utiliser les magasins de fonctionnalités d'apprentissage automatique

L'exemple de code suivant illustre comment :

- Définir des autorisations IAM
- Création d'un rôle SageMaker d'exécution
- Création de groupes de fonctionnalités
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Amazon SageMaker Feature Store Tutorial Script - Version 3
# This script demonstrates how to use Amazon SageMaker Feature Store with AWS CLI

# Setup logging
LOG_FILE="sagemaker-featurestore-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting SageMaker Feature Store tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"
echo ""

# Track created resources for cleanup
CREATED_RESOURCES=()

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
```

```
    exit 1
}

# Function to check command status
check_status() {
    if echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to wait for feature group to be created
wait_for_feature_group() {
    local feature_group_name=$1
    local status="Creating"

    echo "Waiting for feature group ${feature_group_name} to be created..."

    while [ "$status" = "Creating" ]; do
        sleep 5
        status=$(aws sagemaker describe-feature-group \
            --feature-group-name "${feature_group_name}" \
            --query 'FeatureGroupStatus' \
            --output text)
        echo "Current status: ${status}"

        if [ "$status" = "Failed" ]; then
            handle_error "Feature group ${feature_group_name} creation failed"
        fi
    done

    echo "Feature group ${feature_group_name} is now ${status}"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Clean up in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[$i]}"
        resource_type=$(echo "$resource" | cut -d: -f1)
        resource_name=$(echo "$resource" | cut -d: -f2)

        echo "Deleting $resource_type: $resource_name"
```

```
        case "$resource_type" in
            "FeatureGroup")
                aws sagemaker delete-feature-group --feature-group-name
 "$resource_name"
                ;;
            "S3Bucket")
                echo "Emptying S3 bucket: $resource_name"
                aws s3 rm "s3://$resource_name" --recursive 2>/dev/null
                echo "Deleting S3 bucket: $resource_name"
                aws s3api delete-bucket --bucket "$resource_name" 2>/dev/null
                ;;
            "IAMRole")
                echo "Detaching policies from role: $resource_name"
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>/dev/null
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>/dev/null
                echo "Deleting IAM role: $resource_name"
                aws iam delete-role --role-name "$resource_name" 2>/dev/null
                ;;
            *)
                echo "Unknown resource type: $resource_type"
                ;;
        esac
    done
}

# Function to create SageMaker execution role
create_sagemaker_role() {
    local role_name="SageMakerFeatureStoreRole-$(openssl rand -hex 4)"

    echo "Creating SageMaker execution role: $role_name" >&2

    # Create trust policy document
    local trust_policy='{
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "Service": "sagemaker.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
```

```
            }
        ]
    }'

    # Create the role
    local role_result=$(aws iam create-role \
        --role-name "$role_name" \
        --assume-role-policy-document "$trust_policy" \
        --description "SageMaker execution role for Feature Store tutorial" 2>&1)

    if echo "$role_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to create IAM role: $role_result"
    fi

    echo "Role created successfully" >&2
    CREATED_RESOURCES+=("IAMRole:$role_name")

    # Tag the role
    echo "Tagging IAM role..." >&2
    aws iam tag-role --role-name "$role_name" --tags Key=project,Value=doc-smith
Key=tutorial,Value=sagemaker-featurestore 2>&1

    # Attach necessary policies
    echo "Attaching policies to role..." >&2

    # SageMaker execution policy
    local policy1_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>&1)

    if echo "$policy1_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to attach SageMaker policy: $policy1_result"
    fi

    # S3 access policy
    local policy2_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>&1)

    if echo "$policy2_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to attach S3 policy: $policy2_result"
    fi

    # Get account ID for role ARN
```

```
    local account_id=$(aws sts get-caller-identity --query Account --output text)
    local role_arn="arn:aws:iam::${account_id}:role/${role_name}"

    echo "Role ARN: $role_arn" >&2
    echo "Waiting 10 seconds for role to propagate..." >&2
    sleep 10

    # Return only the role ARN to stdout
    echo "$role_arn"
}

# Handle SageMaker execution role
ROLE_ARN=""

if [ -z "$1" ]; then
    echo "Creating SageMaker execution role automatically..."
    ROLE_ARN=$(create_sagemaker_role)
    if [ -z "$ROLE_ARN" ]; then
        handle_error "Failed to create SageMaker execution role"
    fi
else
    ROLE_ARN="$1"

    # Validate the role ARN
    ROLE_NAME=$(echo "$ROLE_ARN" | sed 's/.*role\///')
    ROLE_CHECK=$(aws iam get-role --role-name "$ROLE_NAME" 2>&1)
    if echo "$ROLE_CHECK" | grep -i "error" > /dev/null; then
        echo "Creating a new role automatically..."
        ROLE_ARN=$(create_sagemaker_role)
        if [ -z "$ROLE_ARN" ]; then
            handle_error "Failed to create SageMaker execution role"
        fi
    fi
fi

# Handle cleanup option
AUTO_CLEANUP=""
if [ -n "$2" ]; then
    AUTO_CLEANUP="$2"
fi

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
echo "Using random identifier: $RANDOM_ID"
```

```
# Set variables
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
check_status "$ACCOUNT_ID"
echo "Account ID: $ACCOUNT_ID"

# Get current region
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No default region configured, using: $REGION"
else
    echo "Using region: $REGION"
fi
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="sagemaker-featurestore-${RANDOM_ID}-${ACCOUNT_ID}"
fi
PREFIX="featurestore-tutorial"
CURRENT_TIME=$(date +%s)

echo "Creating S3 bucket: $S3_BUCKET_NAME"
# Create bucket in current region (skip if using shared bucket)
if [ "$BUCKET_IS_SHARED" = "false" ]; then
if [ "$REGION" = "us-east-1" ]; then
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" 2>&1)
else
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" \
        --create-bucket-configuration LocationConstraint="$REGION" 2>&1)
fi

if echo "$BUCKET_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to create S3 bucket: $BUCKET_RESULT"
```

```
    exit 1
fi

echo "$BUCKET_RESULT"
CREATED_RESOURCES+=("S3Bucket:$S3_BUCKET_NAME")

# Tag the S3 bucket
echo "Tagging S3 bucket: $S3_BUCKET_NAME"
aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=sagemaker-
featurestore}]' 2>&1

# Block public access to the bucket
BLOCK_RESULT=$(aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
 2>&1)

if echo "$BLOCK_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to block public access to S3 bucket: $BLOCK_RESULT"
    cleanup_resources
    exit 1
fi
else
    echo "Using shared bucket (skipping creation)"
fi

# Create feature groups
echo "Creating feature groups..."

# Create customers feature group
CUSTOMERS_FEATURE_GROUP_NAME="customers-feature-group-${RANDOM_ID}"
echo "Creating customers feature group: $CUSTOMERS_FEATURE_GROUP_NAME"

CUSTOMERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "name", "FeatureType": "String"},
        {"FeatureName": "age", "FeatureType": "Integral"},
        {"FeatureName": "address", "FeatureType": "String"},
```

```
        {"FeatureName": "membership_type", "FeatureType": "String"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
        },
        "DisableGlueTableCreation": false
    }' \
    --role-arn "$ROLE_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$CUSTOMERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create customers feature group: $CUSTOMERS_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$CUSTOMERS_FEATURE_GROUP_NAME")

# Create orders feature group
ORDERS_FEATURE_GROUP_NAME="orders-feature-group-${RANDOM_ID}"
echo "Creating orders feature group: $ORDERS_FEATURE_GROUP_NAME"

ORDERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "order_id", "FeatureType": "String"},
        {"FeatureName": "order_date", "FeatureType": "String"},
        {"FeatureName": "product", "FeatureType": "String"},
        {"FeatureName": "quantity", "FeatureType": "Integral"},
        {"FeatureName": "amount", "FeatureType": "Fractional"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
```

```
        },
        "DisableGlueTableCreation": false
    }' \
    --role-arn "$ROLE_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$ORDERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create orders feature group: $ORDERS_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$ORDERS_FEATURE_GROUP_NAME")

# Wait for feature groups to be created
wait_for_feature_group "$CUSTOMERS_FEATURE_GROUP_NAME"
wait_for_feature_group "$ORDERS_FEATURE_GROUP_NAME"

# Ingest data into feature groups
echo "Ingesting data into feature groups..."

# Ingest customer data
echo "Ingesting customer data..."
CUSTOMER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "name", "ValueAsString": "John Doe"},
        {"FeatureName": "age", "ValueAsString": "35"},
        {"FeatureName": "address", "ValueAsString": "123 Main St"},
        {"FeatureName": "membership_type", "ValueAsString": "premium"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 1 data: $CUSTOMER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER1_RESPONSE"
```

```
CUSTOMER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "name", "ValueAsString": "Jane Smith"},
        {"FeatureName": "age", "ValueAsString": "28"},
        {"FeatureName": "address", "ValueAsString": "456 Oak Ave"},
        {"FeatureName": "membership_type", "ValueAsString": "standard"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 2 data: $CUSTOMER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER2_RESPONSE"

# Ingest order data
echo "Ingesting order data..."
ORDER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-001"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-15"},
        {"FeatureName": "product", "ValueAsString": "Laptop"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "1299.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 1 data: $ORDER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER1_RESPONSE"

ORDER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
```

```
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-002"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-20"},
        {"FeatureName": "product", "ValueAsString": "Smartphone"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "899.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 2 data: $ORDER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER2_RESPONSE"

# Retrieve records from feature groups
echo "Retrieving records from feature groups..."

# Get a single customer record
echo "Getting customer record with ID 573291:"
CUSTOMER_RECORD=$(aws sagemaker-featurestore-runtime get-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-value-as-string "573291" 2>&1)

if echo "$CUSTOMER_RECORD" | grep -i "error" > /dev/null; then
    echo "Failed to get customer record: $CUSTOMER_RECORD"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER_RECORD"

# Get multiple records using batch-get-record
echo "Getting multiple records using batch-get-record:"
BATCH_RECORDS=$(aws sagemaker-featurestore-runtime batch-get-record \
    --identifiers '[
        {
            "FeatureGroupName": "'${CUSTOMERS_FEATURE_GROUP_NAME}'",
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        },
        {
            "FeatureGroupName": "'${ORDERS_FEATURE_GROUP_NAME}'",
```

```
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        }
    ]' 2>&1)

if echo "$BATCH_RECORDS" | grep -i "error" > /dev/null && ! echo "$BATCH_RECORDS"
 | grep -i "Records" > /dev/null; then
    echo "Failed to get batch records: $BATCH_RECORDS"
    cleanup_resources
    exit 1
fi

echo "$BATCH_RECORDS"

# List feature groups
echo "Listing feature groups:"
FEATURE_GROUPS=$(aws sagemaker list-feature-groups 2>&1)

if echo "$FEATURE_GROUPS" | grep -i "error" > /dev/null; then
    echo "Failed to list feature groups: $FEATURE_GROUPS"
    cleanup_resources
    exit 1
fi

echo "$FEATURE_GROUPS"

# Display summary of created resources
echo ""
echo "============================================="
echo "TUTORIAL COMPLETED SUCCESSFULLY!"
echo "============================================="
echo "Resources created:"
echo "- S3 Bucket: $S3_BUCKET_NAME"
echo "- Customers Feature Group: $CUSTOMERS_FEATURE_GROUP_NAME"
echo "- Orders Feature Group: $ORDERS_FEATURE_GROUP_NAME"
if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
    echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o 'IAMRole:
[^[:space:]]*' | cut -d: -f2)"
fi
echo ""
echo "You can now:"
echo "1. View your feature groups in the SageMaker console"
echo "2. Query the offline store using Amazon Athena"
echo "3. Use the feature groups in your ML workflows"
echo "============================================="
```

```
echo ""

# Handle cleanup
if [ "$AUTO_CLEANUP" = "y" ]; then
    echo "Auto-cleanup enabled. Starting cleanup..."
    cleanup_resources
    echo "Cleanup completed."
elif [ "$AUTO_CLEANUP" = "n" ]; then
    echo "Auto-cleanup disabled. Resources will remain in your account."
    echo "To clean up later, run this script again with cleanup option 'y'"
else
    echo "==========================================="
    echo "CLEANUP CONFIRMATION"
    echo "==========================================="
    echo "Do you want to clean up all created resources? (y/n): "
    read -r CLEANUP_CHOICE

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        echo "Starting cleanup..."
        cleanup_resources
        echo "Cleanup completed."
    else
        echo "Skipping cleanup. Resources will remain in your account."
        echo "To clean up later, delete the following resources:"
        echo "- Feature Groups: $CUSTOMERS_FEATURE_GROUP_NAME,
 $ORDERS_FEATURE_GROUP_NAME"
        echo "- S3 Bucket: $S3_BUCKET_NAME"
        if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
            echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o
 'IAMRole:[^[:space:]]*' | cut -d: -f2)"
        fi
        echo ""
        echo "Estimated ongoing cost: ~$0.01 per month for online store"
    fi
fi

echo "Script completed at $(date)"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreateBucket

- CreateFeatureGroup
- CreateRole
- DeleteBucket
- DeleteFeatureGroup
- DeleteRole
- DescribeFeatureGroup
- DetachRolePolicy
- GetCallerIdentity
- GetRole
- ListFeatureGroups
- PutPublicAccessBlock
- Rm

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Commencer à utiliser les clusters Kubernetes gérés

L'exemple de code suivant illustre comment :

- Créez un VPC pour votre cluster EKS
- Créez des rôles IAM pour votre cluster EKS
- Créer votre cluster EKS
- Configurer kubectl pour communiquer avec votre cluster
- Créer un groupe de nœuds géré
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Amazon EKS Cluster Creation Script (v2)
# This script creates an Amazon EKS cluster with a managed node group using the
 AWS CLI

# Set up logging
LOG_FILE="eks-cluster-creation-v2.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon EKS cluster creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Error handling function
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check command success
check_command() {
    if [ $? -ne 0 ] || echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to check if kubectl is installed
check_kubectl() {
    if ! command -v kubectl &> /dev/null; then
        echo "WARNING: kubectl is not installed or not in your PATH."
```

```
        echo ""
        echo "To install kubectl, follow these instructions based on your
 operating system:"
        echo ""
        echo "For Linux:"
        echo "  1. Download the latest release:"
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl\""
        echo ""
        echo "  2. Make the kubectl binary executable:"
        echo "     chmod +x ./kubectl"
        echo ""
        echo "  3. Move the binary to your PATH:"
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For macOS:"
        echo "  1. Using Homebrew:"
        echo "     brew install kubectl"
        echo "     or"
        echo "  2. Using curl:"
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/darwin/amd64/kubectl\""
        echo "     chmod +x ./kubectl"
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For Windows:"
        echo "  1. Using curl:"
        echo "     curl -LO \"https://dl.k8s.io/release/v1.28.0/bin/windows/
amd64/kubectl.exe\""
        echo "     Add the binary to your PATH"
        echo "     or"
        echo "  2. Using Chocolatey:"
        echo "     choco install kubernetes-cli"
        echo ""
        echo "After installation, verify with: kubectl version --client"
        echo ""
        return 1
    fi
    return 0
}

# Generate a random identifier for resource names
RANDOM_ID=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 6 | head -n 1)
STACK_NAME="eks-vpc-stack-${RANDOM_ID}"
```

```
CLUSTER_NAME="eks-cluster-${RANDOM_ID}"
NODEGROUP_NAME="eks-nodegroup-${RANDOM_ID}"
CLUSTER_ROLE_NAME="EKSClusterRole-${RANDOM_ID}"
NODE_ROLE_NAME="EKSNodeRole-${RANDOM_ID}"

echo "Using the following resource names:"
echo "- VPC Stack: $STACK_NAME"
echo "- EKS Cluster: $CLUSTER_NAME"
echo "- Node Group: $NODEGROUP_NAME"
echo "- Cluster IAM Role: $CLUSTER_ROLE_NAME"
echo "- Node IAM Role: $NODE_ROLE_NAME"

# Array to track created resources for cleanup
declare -a CREATED_RESOURCES

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources in reverse order..."

    # Check if node group exists and delete it
    if aws eks list-nodegroups --cluster-name "$CLUSTER_NAME" --query
 "nodegroups[?contains(@,'$NODEGROUP_NAME')]" --output text 2>/dev/null | grep -q
 "$NODEGROUP_NAME"; then
        echo "Deleting node group: $NODEGROUP_NAME"
        aws eks delete-nodegroup --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
        echo "Waiting for node group deletion to complete..."
        aws eks wait nodegroup-deleted --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME"
        echo "Node group deleted successfully."
    fi

    # Check if cluster exists and delete it
    if aws eks describe-cluster --name "$CLUSTER_NAME" 2>/dev/null; then
        echo "Deleting cluster: $CLUSTER_NAME"
        aws eks delete-cluster --name "$CLUSTER_NAME"
        echo "Waiting for cluster deletion to complete (this may take several
 minutes)..."
        aws eks wait cluster-deleted --name "$CLUSTER_NAME"
        echo "Cluster deleted successfully."
    fi

    # Check if CloudFormation stack exists and delete it
```

```
    if aws cloudformation describe-stacks --stack-name "$STACK_NAME" 2>/dev/null;
 then
        echo "Deleting CloudFormation stack: $STACK_NAME"
        aws cloudformation delete-stack --stack-name "$STACK_NAME"
        echo "Waiting for CloudFormation stack deletion to complete..."
        aws cloudformation wait stack-delete-complete --stack-name "$STACK_NAME"
        echo "CloudFormation stack deleted successfully."
    fi

    # Clean up IAM roles
    if aws iam get-role --role-name "$NODE_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from node role: $NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSWorkerNodePolicy --role-name "$NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEC2ContainerRegistryReadOnly --role-name "$NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKS_CNI_Policy --role-name "$NODE_ROLE_NAME"
        echo "Deleting node role: $NODE_ROLE_NAME"
        aws iam delete-role --role-name "$NODE_ROLE_NAME"
        echo "Node role deleted successfully."
    fi

    if aws iam get-role --role-name "$CLUSTER_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from cluster role: $CLUSTER_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name "$CLUSTER_ROLE_NAME"
        echo "Deleting cluster role: $CLUSTER_ROLE_NAME"
        aws iam delete-role --role-name "$CLUSTER_ROLE_NAME"
        echo "Cluster role deleted successfully."
    fi

    echo "Cleanup complete."
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Cleaning up resources..."; cleanup_resources;
 exit 1' SIGINT SIGTERM

# Verify AWS CLI configuration
echo "Verifying AWS CLI configuration..."
AWS_ACCOUNT_INFO=$(aws sts get-caller-identity)
check_command "$AWS_ACCOUNT_INFO"
echo "AWS CLI is properly configured."
```

```
# Step 1: Create VPC using CloudFormation
echo "Step 1: Creating VPC with CloudFormation..."
echo "Creating CloudFormation stack: $STACK_NAME"

# Create the CloudFormation stack
CF_CREATE_OUTPUT=$(aws cloudformation create-stack \
  --stack-name "$STACK_NAME" \
  --template-url https://s3.us-west-2.amazonaws.com/amazon-eks/
cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs)
check_command "$CF_CREATE_OUTPUT"
CREATED_RESOURCES+=("CloudFormation Stack: $STACK_NAME")

echo "Waiting for CloudFormation stack to complete (this may take a few
 minutes)..."
aws cloudformation wait stack-create-complete --stack-name "$STACK_NAME"
if [ $? -ne 0 ]; then
    handle_error "CloudFormation stack creation failed"
fi
echo "CloudFormation stack created successfully."

# Step 2: Create IAM roles for EKS
echo "Step 2: Creating IAM roles for EKS..."

# Create cluster role trust policy
echo "Creating cluster role trust policy..."
cat > eks-cluster-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create cluster role
echo "Creating cluster IAM role: $CLUSTER_ROLE_NAME"
```

```
CLUSTER_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$CLUSTER_ROLE_NAME" \
  --assume-role-policy-document file://"eks-cluster-role-trust-policy.json")
check_command "$CLUSTER_ROLE_OUTPUT"
aws iam tag-role --role-name "$CLUSTER_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $CLUSTER_ROLE_NAME")

# Attach policy to cluster role
echo "Attaching EKS cluster policy to role..."
ATTACH_CLUSTER_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy \
  --role-name "$CLUSTER_ROLE_NAME")
check_command "$ATTACH_CLUSTER_POLICY_OUTPUT"

# Create node role trust policy
echo "Creating node role trust policy..."
cat > node-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create node role
echo "Creating node IAM role: $NODE_ROLE_NAME"
NODE_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$NODE_ROLE_NAME" \
  --assume-role-policy-document file://"node-role-trust-policy.json")
check_command "$NODE_ROLE_OUTPUT"
aws iam tag-role --role-name "$NODE_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $NODE_ROLE_NAME")

# Attach policies to node role
echo "Attaching EKS node policies to role..."
```

```
ATTACH_NODE_POLICY1_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY1_OUTPUT"

ATTACH_NODE_POLICY2_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY2_OUTPUT"

ATTACH_NODE_POLICY3_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY3_OUTPUT"

# Step 3: Get VPC and subnet information
echo "Step 3: Getting VPC and subnet information..."

VPC_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='VpcId'].OutputValue" \
  --output text)
if [ -z "$VPC_ID" ]; then
    handle_error "Failed to get VPC ID from CloudFormation stack"
fi
echo "VPC ID: $VPC_ID"

SUBNET_IDS=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SubnetIds'].OutputValue" \
  --output text)
if [ -z "$SUBNET_IDS" ]; then
    handle_error "Failed to get Subnet IDs from CloudFormation stack"
fi
echo "Subnet IDs: $SUBNET_IDS"

SECURITY_GROUP_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SecurityGroups'].OutputValue" \
  --output text)
if [ -z "$SECURITY_GROUP_ID" ]; then
    handle_error "Failed to get Security Group ID from CloudFormation stack"
fi
echo "Security Group ID: $SECURITY_GROUP_ID"
```

```
# Step 4: Create EKS cluster
echo "Step 4: Creating EKS cluster: $CLUSTER_NAME"

CLUSTER_ROLE_ARN=$(aws iam get-role --role-name "$CLUSTER_ROLE_NAME" --query
 "Role.Arn" --output text)
if [ -z "$CLUSTER_ROLE_ARN" ]; then
    handle_error "Failed to get Cluster Role ARN"
fi

echo "Creating EKS cluster (this will take 10-15 minutes)..."
CREATE_CLUSTER_OUTPUT=$(aws eks create-cluster \
  --name "$CLUSTER_NAME" \
  --role-arn "$CLUSTER_ROLE_ARN" \
  --resources-vpc-config
 subnetIds="$SUBNET_IDS",securityGroupIds="$SECURITY_GROUP_ID" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_CLUSTER_OUTPUT"
CREATED_RESOURCES+=("EKS Cluster: $CLUSTER_NAME")

echo "Waiting for EKS cluster to become active (this may take 10-15 minutes)..."
aws eks wait cluster-active --name "$CLUSTER_NAME"
if [ $? -ne 0 ]; then
    handle_error "Cluster creation failed or timed out"
fi
echo "EKS cluster is now active."

# Step 5: Configure kubectl
echo "Step 5: Configuring kubectl to communicate with the cluster..."

# Check if kubectl is installed
if ! check_kubectl; then
    echo "Will skip kubectl configuration steps but continue with the script."
    echo "You can manually configure kubectl later with: aws eks update-
kubeconfig --name \"$CLUSTER_NAME\""
else
    UPDATE_KUBECONFIG_OUTPUT=$(aws eks update-kubeconfig --name "$CLUSTER_NAME")
    check_command "$UPDATE_KUBECONFIG_OUTPUT"
    echo "kubectl configured successfully."

    # Test kubectl configuration
    echo "Testing kubectl configuration..."
    KUBECTL_TEST_OUTPUT=$(kubectl get svc 2>&1)
    if [ $? -ne 0 ]; then
```

```
        echo "Warning: kubectl configuration test failed. This might be due to
 permissions or network issues."
        echo "Error details: $KUBECTL_TEST_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$KUBECTL_TEST_OUTPUT"
        echo "kubectl configuration test successful."
    fi
fi

# Step 6: Create managed node group
echo "Step 6: Creating managed node group: $NODEGROUP_NAME"

NODE_ROLE_ARN=$(aws iam get-role --role-name "$NODE_ROLE_NAME" --query "Role.Arn"
 --output text)
if [ -z "$NODE_ROLE_ARN" ]; then
    handle_error "Failed to get Node Role ARN"
fi

# Convert comma-separated subnet IDs to space-separated for the create-nodegroup
 command
SUBNET_IDS_ARRAY=(${SUBNET_IDS//,/ })

echo "Creating managed node group (this will take 5-10 minutes)..."
CREATE_NODEGROUP_OUTPUT=$(aws eks create-nodegroup \
  --cluster-name "$CLUSTER_NAME" \
  --nodegroup-name "$NODEGROUP_NAME" \
  --node-role "$NODE_ROLE_ARN" \
  --subnets "${SUBNET_IDS_ARRAY[@]}" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_NODEGROUP_OUTPUT"
CREATED_RESOURCES+=("EKS Node Group: $NODEGROUP_NAME")

echo "Waiting for node group to become active (this may take 5-10 minutes)..."
aws eks wait nodegroup-active --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
if [ $? -ne 0 ]; then
    handle_error "Node group creation failed or timed out"
fi
echo "Node group is now active."

# Step 7: Verify nodes
echo "Step 7: Verifying nodes..."
```

```
echo "Waiting for nodes to register with the cluster (this may take a few
 minutes)..."
sleep 60  # Give nodes more time to register

# Check if kubectl is installed before attempting to use it
if ! check_kubectl; then
    echo "Cannot verify nodes without kubectl. Skipping this step."
    echo "You can manually verify nodes after installing kubectl with: kubectl
 get nodes"
else
    NODES_OUTPUT=$(kubectl get nodes 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: Unable to get nodes. This might be due to permissions or
 the nodes are still registering."
        echo "Error details: $NODES_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$NODES_OUTPUT"
        echo "Nodes verified successfully."
    fi
fi

# Step 8: View resources
echo "Step 8: Viewing cluster resources..."

echo "Cluster information:"
CLUSTER_INFO=$(aws eks describe-cluster --name "$CLUSTER_NAME")
echo "$CLUSTER_INFO"

echo "Node group information:"
NODEGROUP_INFO=$(aws eks describe-nodegroup --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME")
echo "$NODEGROUP_INFO"

echo "Kubernetes resources:"
if ! check_kubectl; then
    echo "Cannot list Kubernetes resources without kubectl. Skipping this step."
    echo "You can manually list resources after installing kubectl with: kubectl
 get all --all-namespaces"
else
    KUBE_RESOURCES=$(kubectl get all --all-namespaces 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: Unable to get Kubernetes resources. This might be due to
 permissions."
```

```
        echo "Error details: $KUBE_RESOURCES"
        echo "Continuing with script execution..."
    else
        echo "$KUBE_RESOURCES"
    fi
fi

# Display summary of created resources
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource"
done
echo "==========================================="

# Prompt for cleanup
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    cleanup_resources
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run the following commands:"
    echo "1. Delete node group: aws eks delete-nodegroup --cluster-name
 $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "2. Wait for node group deletion: aws eks wait nodegroup-deleted --
cluster-name $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "3. Delete cluster: aws eks delete-cluster --name $CLUSTER_NAME"
    echo "4. Wait for cluster deletion: aws eks wait cluster-deleted --name
 $CLUSTER_NAME"
    echo "5. Delete CloudFormation stack: aws cloudformation delete-stack --
stack-name $STACK_NAME"
    echo "6. Detach and delete IAM roles for the node group and cluster"
fi

echo "Script completed at $(date)"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreateCluster
  - CreateNodegroup
  - CreateRole
  - CreateStack
  - DeleteCluster
  - DeleteNodegroup
  - DeleteRole
  - DeleteStack
  - DescribeCluster
  - DescribeNodegroup
  - DescribeStacks
  - DetachRolePolicy
  - GetCallerIdentity
  - GetRole
  - ListNodegroups
  - UpdateKubeconfig
  - Attendre (CloudFormation)
  - Attendre (Amazon EKS)

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Débuter avec le streaming géré

L'exemple de code suivant illustre comment :

- Création d'un cluster MSK

- Création d'autorisations IAM pour l'accès MSK
- Création d'une machine cliente
- Trouvez des courtiers Bootstrap
- Configuration de la machine cliente
- Création d'un sujet et de produce/consume données
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Amazon MSK Getting Started Tutorial Script - Version 8
# This script automates the steps in the Amazon MSK Getting Started tutorial
# It creates an MSK cluster, sets up IAM permissions, creates a client machine,
# and configures the client to interact with the cluster

# Set up logging
LOG_FILE="msk_tutorial_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon MSK Getting Started Tutorial Script - Version 8"
echo "Logging to $LOG_FILE"
echo "================================================"

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created so far:"
    if [ -n "$CLUSTER_ARN" ]; then echo "- MSK Cluster: $CLUSTER_ARN"; fi
    if [ -n "$POLICY_ARN" ]; then echo "- IAM Policy: $POLICY_ARN"; fi
    if [ -n "$ROLE_NAME" ]; then echo "- IAM Role: $ROLE_NAME"; fi
```

```
    if [ -n "$INSTANCE_PROFILE_NAME" ]; then echo "- IAM Instance Profile:
 $INSTANCE_PROFILE_NAME"; fi
    if [ -n "$CLIENT_SG_ID" ]; then echo "- Client Security Group:
 $CLIENT_SG_ID"; fi
    if [ -n "$INSTANCE_ID" ]; then echo "- EC2 Instance: $INSTANCE_ID"; fi
    if [ -n "$KEY_NAME" ]; then echo "- Key Pair: $KEY_NAME"; fi

    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check if a resource exists
resource_exists() {
    local resource_type="$1"
    local resource_id="$2"

    case "$resource_type" in
        "cluster")
            aws kafka describe-cluster --cluster-arn "$resource_id" &>/dev/null
            ;;
        "policy")
            aws iam get-policy --policy-arn "$resource_id" &>/dev/null
            ;;
        "role")
            aws iam get-role --role-name "$resource_id" &>/dev/null
            ;;
        "instance-profile")
            aws iam get-instance-profile --instance-profile-name "$resource_id"
 &>/dev/null
            ;;
        "security-group")
            aws ec2 describe-security-groups --group-ids "$resource_id" &>/dev/
null
            ;;
        "instance")
            aws ec2 describe-instances --instance-ids "$resource_id" --query
 'Reservations[0].Instances[0].State.Name' --output text | grep -v "terminated"
 &>/dev/null
            ;;
        "key-pair")
            aws ec2 describe-key-pairs --key-names "$resource_id" &>/dev/null
            ;;
    esac
```

```
}

# Function to remove security group references
remove_security_group_references() {
    local sg_id="$1"

    if [ -z "$sg_id" ]; then
        echo "No security group ID provided for reference removal"
        return
    fi

    echo "Removing security group references for $sg_id"

    # Get all security groups in the VPC that might reference our client security
 group
    local vpc_security_groups=$(aws ec2 describe-security-groups \
        --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" \
        --query 'SecurityGroups[].GroupId' \
        --output text 2>/dev/null)

    if [ -n "$vpc_security_groups" ]; then
        for other_sg in $vpc_security_groups; do
            if [ "$other_sg" != "$sg_id" ]; then
                echo "Checking security group $other_sg for references to $sg_id"

                # Get the security group details in JSON format
                local sg_details=$(aws ec2 describe-security-groups \
                    --group-ids "$other_sg" \
                    --output json 2>/dev/null)

                if [ -n "$sg_details" ]; then
                    # Check if our security group is referenced in inbound rules
                    local has_inbound_ref=$(echo "$sg_details" | grep -o
 "\"GroupId\": \"$sg_id\"" | head -1)

                    if [ -n "$has_inbound_ref" ]; then
                        echo "Found inbound rules in $other_sg referencing
 $sg_id, removing them..."

                        # Try to remove common rule types
                        echo "Attempting to remove all-traffic rule"
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol all \
```

```
                            --source-group "$sg_id" 2>/dev/null || echo "No all-
traffic rule to remove"

                        # Try to remove TCP rules on common ports
                        for port in 22 80 443 9092 9094 9096; do
                            aws ec2 revoke-security-group-ingress \
                                --group-id "$other_sg" \
                                --protocol tcp \
                                --port "$port" \
                                --source-group "$sg_id" 2>/dev/null || true
                        done

                        # Try to remove UDP rules
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol udp \
                            --source-group "$sg_id" 2>/dev/null || true
                    fi

                    # Check for outbound rules (less common but possible)
                    local has_outbound_ref=$(echo "$sg_details" | grep -A 20
 "IpPermissionsEgress" | grep -o "\"GroupId\": \"$sg_id\"" | head -1)

                    if [ -n "$has_outbound_ref" ]; then
                        echo "Found outbound rules in $other_sg referencing
 $sg_id, removing them..."

                        aws ec2 revoke-security-group-egress \
                            --group-id "$other_sg" \
                            --protocol all \
                            --source-group "$sg_id" 2>/dev/null || echo "No
 outbound all-traffic rule to remove"
                    fi
                fi
            fi
        done
    fi

    echo "Completed security group reference removal for $sg_id"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."
```

```
   # Delete EC2 instance if it exists
   if [ -n "$INSTANCE_ID" ] && resource_exists "instance" "$INSTANCE_ID"; then
       echo "Terminating EC2 instance: $INSTANCE_ID"
       aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || echo "Failed
to terminate instance"
       echo "Waiting for instance to terminate..."
       aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID" || echo
"Failed to wait for instance termination"
   fi

   # Delete MSK cluster first (to remove dependencies on security group)
   if [ -n "$CLUSTER_ARN" ] && resource_exists "cluster" "$CLUSTER_ARN"; then
       echo "Deleting MSK cluster: $CLUSTER_ARN"
       aws kafka delete-cluster --cluster-arn "$CLUSTER_ARN" || echo "Failed to
delete cluster"

       # Wait a bit for the cluster deletion to start
       echo "Waiting 30 seconds for cluster deletion to begin..."
       sleep 30
   fi

   # Remove security group references before attempting deletion
   if [ -n "$CLIENT_SG_ID" ] && resource_exists "security-group"
"$CLIENT_SG_ID"; then
       remove_security_group_references "$CLIENT_SG_ID"

       echo "Deleting security group: $CLIENT_SG_ID"
       # Try multiple times with longer delays to ensure dependencies are
removed
       for i in {1..10}; do
           if aws ec2 delete-security-group --group-id "$CLIENT_SG_ID"; then
               echo "Security group deleted successfully"
               break
           fi
           echo "Failed to delete security group (attempt $i/10), retrying in 30
seconds..."
           sleep 30
       done
   fi

   # Delete key pair if it exists
   if [ -n "$KEY_NAME" ] && resource_exists "key-pair" "$KEY_NAME"; then
       echo "Deleting key pair: $KEY_NAME"
```

```
        aws ec2 delete-key-pair --key-name "$KEY_NAME" || echo "Failed to delete
key pair"
    fi

    # Remove role from instance profile
    if [ -n "$ROLE_NAME" ] && [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists
"instance-profile" "$INSTANCE_PROFILE_NAME"; then
        echo "Removing role from instance profile"
        aws iam remove-role-from-instance-profile \
            --instance-profile-name "$INSTANCE_PROFILE_NAME" \
            --role-name "$ROLE_NAME" || echo "Failed to remove role from instance
profile"
    fi

    # Delete instance profile
    if [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists "instance-profile"
"$INSTANCE_PROFILE_NAME"; then
        echo "Deleting instance profile: $INSTANCE_PROFILE_NAME"
        aws iam delete-instance-profile \
            --instance-profile-name "$INSTANCE_PROFILE_NAME" || echo "Failed to
delete instance profile"
    fi

    # Detach policy from role
    if [ -n "$ROLE_NAME" ] && [ -n "$POLICY_ARN" ] && resource_exists "role"
"$ROLE_NAME"; then
        echo "Detaching policy from role"
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "$POLICY_ARN" || echo "Failed to detach policy"
    fi

    # Delete role
    if [ -n "$ROLE_NAME" ] && resource_exists "role" "$ROLE_NAME"; then
        echo "Deleting role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" || echo "Failed to delete
role"
    fi

    # Delete policy
    if [ -n "$POLICY_ARN" ] && resource_exists "policy" "$POLICY_ARN"; then
        echo "Deleting policy: $POLICY_ARN"
        aws iam delete-policy --policy-arn "$POLICY_ARN" || echo "Failed to
delete policy"
```

```
    fi

    echo "Cleanup completed"
}

# Function to find a suitable subnet and instance type combination
find_suitable_subnet_and_instance_type() {
    local vpc_id="$1"
    local -a subnet_array=("${!2}")

    # List of instance types to try, in order of preference
    local instance_types=("t3.micro" "t2.micro" "t3.small" "t2.small")

    echo "Finding suitable subnet and instance type combination..."

    for instance_type in "${instance_types[@]}"; do
        echo "Trying instance type: $instance_type"

        for subnet_id in "${subnet_array[@]}"; do
            # Get the availability zone for this subnet
            local az=$(aws ec2 describe-subnets \
                --subnet-ids "$subnet_id" \
                --query 'Subnets[0].AvailabilityZone' \
                --output text)

            echo "  Checking subnet $subnet_id in AZ $az"

            # Check if this instance type is available in this AZ
            local available=$(aws ec2 describe-instance-type-offerings \
                --location-type availability-zone \
                --filters "Name=location,Values=$az" "Name=instance-type,Values=
$instance_type" \
                --query 'InstanceTypeOfferings[0].InstanceType' \
                --output text 2>/dev/null)

            if [ "$available" = "$instance_type" ]; then
                echo "  # Found suitable combination: $instance_type in $az
 (subnet: $subnet_id)"
                SELECTED_SUBNET_ID="$subnet_id"
                SELECTED_INSTANCE_TYPE="$instance_type"
                return 0
            else
                echo "  # $instance_type not available in $az"
            fi
```

```
        done
    done

    echo "ERROR: Could not find any suitable subnet and instance type
 combination"
    return 1
}

# Generate unique identifiers
RANDOM_SUFFIX=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 8 | head -n 1)
CLUSTER_NAME="MSKTutorialCluster-${RANDOM_SUFFIX}"
POLICY_NAME="msk-tutorial-policy-${RANDOM_SUFFIX}"
ROLE_NAME="msk-tutorial-role-${RANDOM_SUFFIX}"
INSTANCE_PROFILE_NAME="msk-tutorial-profile-${RANDOM_SUFFIX}"
SG_NAME="MSKClientSecurityGroup-${RANDOM_SUFFIX}"

echo "Using the following resource names:"
echo "- Cluster Name: $CLUSTER_NAME"
echo "- Policy Name: $POLICY_NAME"
echo "- Role Name: $ROLE_NAME"
echo "- Instance Profile Name: $INSTANCE_PROFILE_NAME"
echo "- Security Group Name: $SG_NAME"
echo "================================================"

# Step 1: Create an MSK Provisioned cluster
echo "Step 1: Creating MSK Provisioned cluster"

# Get the default VPC ID first
echo "Getting default VPC..."
DEFAULT_VPC_ID=$(aws ec2 describe-vpcs \
    --filters "Name=is-default,Values=true" \
    --query "Vpcs[0].VpcId" \
    --output text)

if [ -z "$DEFAULT_VPC_ID" ] || [ "$DEFAULT_VPC_ID" = "None" ]; then
    handle_error "Could not find default VPC. Please ensure you have a default
 VPC in your region."
fi

echo "Default VPC ID: $DEFAULT_VPC_ID"

# Get available subnets in the default VPC
echo "Getting available subnets in the default VPC..."
SUBNETS=$(aws ec2 describe-subnets \
```

```
    --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" "Name=default-for-
az,Values=true" \
    --query "Subnets[0:3].SubnetId" \
    --output text)

# Convert space-separated subnet IDs to an array
read -r -a SUBNET_ARRAY <<< "$SUBNETS"

if [ ${#SUBNET_ARRAY[@]} -lt 3 ]; then
    handle_error "Not enough subnets available in the default VPC. Need at least
 3 subnets, found ${#SUBNET_ARRAY[@]}."
fi

# Get default security group for the default VPC
echo "Getting default security group for the default VPC..."
DEFAULT_SG=$(aws ec2 describe-security-groups \
    --filters "Name=group-name,Values=default" "Name=vpc-id,Values=
$DEFAULT_VPC_ID" \
    --query "SecurityGroups[0].GroupId" \
    --output text)

if [ -z "$DEFAULT_SG" ] || [ "$DEFAULT_SG" = "None" ]; then
    handle_error "Could not find default security group for VPC $DEFAULT_VPC_ID"
fi

echo "Creating MSK cluster: $CLUSTER_NAME"
echo "Using VPC: $DEFAULT_VPC_ID"
echo "Using subnets: ${SUBNET_ARRAY[0]} ${SUBNET_ARRAY[1]} ${SUBNET_ARRAY[2]}"
echo "Using security group: $DEFAULT_SG"

# Create the MSK cluster with proper error handling
CLUSTER_RESPONSE=$(aws kafka create-cluster \
    --cluster-name "$CLUSTER_NAME" \
    --broker-node-group-info "{\"InstanceType\": \"kafka.t3.small\",
 \"ClientSubnets\": [\"${SUBNET_ARRAY[0]}\", \"${SUBNET_ARRAY[1]}\",
 \"${SUBNET_ARRAY[2]}\"], \"SecurityGroups\": [\"$DEFAULT_SG\"]}" \
    --kafka-version "3.6.0" \
    --number-of-broker-nodes 3 \
    --encryption-info "{\"EncryptionInTransit\": {\"InCluster\": true,
 \"ClientBroker\": \"TLS\"}}" \
    --tags project=doc-smith,tutorial=amazon-managed-streaming-for-apache-kafka-
gs 2>&1)

# Check if the command was successful
```

```
if [ $? -ne 0 ]; then
    handle_error "Failed to create MSK cluster: $CLUSTER_RESPONSE"
fi

# Extract the cluster ARN using grep
CLUSTER_ARN=$(echo "$CLUSTER_RESPONSE" | grep -o '"ClusterArn": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLUSTER_ARN" ]; then
    handle_error "Failed to extract cluster ARN from response: $CLUSTER_RESPONSE"
fi

echo "MSK cluster creation initiated. ARN: $CLUSTER_ARN"
echo "Waiting for cluster to become active (this may take 15-20 minutes)..."

# Wait for the cluster to become active
while true; do
    CLUSTER_STATUS=$(aws kafka describe-cluster --cluster-arn "$CLUSTER_ARN" --
query "ClusterInfo.State" --output text 2>/dev/null)

    if [ $? -ne 0 ]; then
        echo "Failed to get cluster status. Retrying in 30 seconds..."
        sleep 30
        continue
    fi

    echo "Current cluster status: $CLUSTER_STATUS"

    if [ "$CLUSTER_STATUS" = "ACTIVE" ]; then
        echo "Cluster is now active!"
        break
    elif [ "$CLUSTER_STATUS" = "FAILED" ]; then
        handle_error "Cluster creation failed"
    fi

    echo "Still waiting for cluster to become active... (checking again in 60
 seconds)"
    sleep 60
done

echo "============================================="
# Step 2: Create an IAM role granting access to create topics on the Amazon MSK
 cluster
echo "Step 2: Creating IAM policy and role"
```

```
# Get account ID and region
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION=$(aws ec2 describe-availability-zones --query
 'AvailabilityZones[0].RegionName' --output text)
fi

if [ -z "$ACCOUNT_ID" ] || [ -z "$REGION" ]; then
    handle_error "Could not determine AWS account ID or region"
fi

echo "Account ID: $ACCOUNT_ID"
echo "Region: $REGION"

# Create IAM policy
echo "Creating IAM policy: $POLICY_NAME"
POLICY_DOCUMENT="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:Connect\",
                \"kafka-cluster:AlterCluster\",
                \"kafka-cluster:DescribeCluster\"
            ],
            \"Resource\": [
                \"$CLUSTER_ARN\"
            ]
        },
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:*Topic*\",
                \"kafka-cluster:WriteData\",
                \"kafka-cluster:ReadData\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:topic/$CLUSTER_NAME/*\"
            ]
        },
        {
```

```
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:AlterGroup\",
                \"kafka-cluster:DescribeGroup\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:group/$CLUSTER_NAME/*\"
            ]
        }
    ]
}"

POLICY_RESPONSE=$(aws iam create-policy \
    --policy-name "$POLICY_NAME" \
    --policy-document "$POLICY_DOCUMENT" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create IAM policy: $POLICY_RESPONSE"
fi

# Extract the policy ARN using grep
POLICY_ARN=$(echo "$POLICY_RESPONSE" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)

if [ -z "$POLICY_ARN" ]; then
    handle_error "Failed to extract policy ARN from response: $POLICY_RESPONSE"
fi

aws iam tag-policy --policy-arn "$POLICY_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs

echo "IAM policy created. ARN: $POLICY_ARN"

# Create IAM role for EC2
echo "Creating IAM role: $ROLE_NAME"
TRUST_POLICY="{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",
\"Principal\":{\"Service\":\"ec2.amazonaws.com\"},\"Action\":\"sts:AssumeRole
\"}]}"

ROLE_RESPONSE=$(aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document "$TRUST_POLICY" 2>&1)
```

```
# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create IAM role: $ROLE_RESPONSE"
fi

echo "IAM role created: $ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs

# Attach policy to role
echo "Attaching policy to role"
ATTACH_RESPONSE=$(aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "$POLICY_ARN" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to attach policy to role: $ATTACH_RESPONSE"
fi

echo "Policy attached to role"

# Create instance profile and add role to it
echo "Creating instance profile: $INSTANCE_PROFILE_NAME"
PROFILE_RESPONSE=$(aws iam create-instance-profile \
    --instance-profile-name "$INSTANCE_PROFILE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create instance profile: $PROFILE_RESPONSE"
fi

echo "Instance profile created"

echo "Adding role to instance profile"
ADD_ROLE_RESPONSE=$(aws iam add-role-to-instance-profile \
    --instance-profile-name "$INSTANCE_PROFILE_NAME" \
    --role-name "$ROLE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to add role to instance profile: $ADD_ROLE_RESPONSE"
fi
```

```
echo "Role added to instance profile"

# Wait a moment for IAM propagation
echo "Waiting 10 seconds for IAM propagation..."
sleep 10

echo "============================================="

# Step 3: Create a client machine
echo "Step 3: Creating client machine"

# Find a suitable subnet and instance type combination
if ! find_suitable_subnet_and_instance_type "$DEFAULT_VPC_ID" SUBNET_ARRAY[@];
 then
    handle_error "Could not find a suitable subnet and instance type combination"
fi

echo "Selected subnet: $SELECTED_SUBNET_ID"
echo "Selected instance type: $SELECTED_INSTANCE_TYPE"

# Verify the subnet is in the same VPC we're using
SUBNET_VPC_ID=$(aws ec2 describe-subnets \
    --subnet-ids "$SELECTED_SUBNET_ID" \
    --query 'Subnets[0].VpcId' \
    --output text)

if [ "$SUBNET_VPC_ID" != "$DEFAULT_VPC_ID" ]; then
    handle_error "Subnet VPC ($SUBNET_VPC_ID) does not match default VPC
 ($DEFAULT_VPC_ID)"
fi

echo "VPC ID: $SUBNET_VPC_ID"

# Get security group ID from the MSK cluster
echo "Getting security group ID from the MSK cluster"
MSK_SG_ID=$(aws kafka describe-cluster \
    --cluster-arn "$CLUSTER_ARN" \
    --query 'ClusterInfo.BrokerNodeGroupInfo.SecurityGroups[0]' \
    --output text)

if [ -z "$MSK_SG_ID" ] || [ "$MSK_SG_ID" = "None" ]; then
    handle_error "Failed to get security group ID from cluster"
fi
```

```
echo "MSK security group ID: $MSK_SG_ID"

# Create security group for client
echo "Creating security group for client: $SG_NAME"
CLIENT_SG_RESPONSE=$(aws ec2 create-security-group \
    --group-name "$SG_NAME" \
    --description "Security group for MSK client" \
    --vpc-id "$DEFAULT_VPC_ID" \
    --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs}]' 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create security group: $CLIENT_SG_RESPONSE"
fi

# Extract the security group ID using grep
CLIENT_SG_ID=$(echo "$CLIENT_SG_RESPONSE" | grep -o '"GroupId": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLIENT_SG_ID" ]; then
    handle_error "Failed to extract security group ID from response:
 $CLIENT_SG_RESPONSE"
fi

echo "Client security group created. ID: $CLIENT_SG_ID"

# Allow SSH access to client from your IP only
echo "Getting your public IP address"
MY_IP=$(curl -s https://checkip.amazonaws.com 2>/dev/null)

if [ -z "$MY_IP" ]; then
    echo "Warning: Could not determine your IP address. Using 0.0.0.0/0 (not
 recommended for production)"
    MY_IP="0.0.0.0/0"
else
    MY_IP="$MY_IP/32"
    echo "Your IP address: $MY_IP"
fi

echo "Adding SSH ingress rule to client security group"
SSH_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
```

```
    --group-id "$CLIENT_SG_ID" \
    --protocol tcp \
    --port 22 \
    --cidr "$MY_IP" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add SSH ingress rule: $SSH_RULE_RESPONSE"
    echo "You may need to manually add SSH access to security group
 $CLIENT_SG_ID"
fi

echo "SSH ingress rule added"

# Update MSK security group to allow traffic from client security group
echo "Adding ingress rule to MSK security group to allow traffic from client"
MSK_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
    --group-id "$MSK_SG_ID" \
    --protocol all \
    --source-group "$CLIENT_SG_ID" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add ingress rule to MSK security group:
 $MSK_RULE_RESPONSE"
    echo "You may need to manually add ingress rule to security group $MSK_SG_ID"
fi

echo "Ingress rule added to MSK security group"

# Create key pair
KEY_NAME="MSKKeyPair-${RANDOM_SUFFIX}"
echo "Creating key pair: $KEY_NAME"
KEY_RESPONSE=$(aws ec2 create-key-pair --key-name "$KEY_NAME" --tag-
specifications 'ResourceType=key-pair,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-managed-streaming-for-apache-kafka-gs}]' --query
 'KeyMaterial' --output text 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create key pair: $KEY_RESPONSE"
fi

# Save the private key to a file
```

```
KEY_FILE="${KEY_NAME}.pem"
echo "$KEY_RESPONSE" > "$KEY_FILE"
chmod 400 "$KEY_FILE"

echo "Key pair created and saved to $KEY_FILE"

# Get the latest Amazon Linux 2 AMI
echo "Getting latest Amazon Linux 2 AMI ID"
AMI_ID=$(aws ec2 describe-images \
    --owners amazon \
    --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
    --query "sort_by(Images, &CreationDate)[-1].ImageId" \
    --output text 2>/dev/null)

if [ -z "$AMI_ID" ] || [ "$AMI_ID" = "None" ]; then
    handle_error "Failed to get Amazon Linux 2 AMI ID"
fi

echo "Using AMI ID: $AMI_ID"

# Launch EC2 instance with the selected subnet and instance type
echo "Launching EC2 instance"
echo "Instance type: $SELECTED_INSTANCE_TYPE"
echo "Subnet: $SELECTED_SUBNET_ID"

INSTANCE_RESPONSE=$(aws ec2 run-instances \
    --image-id "$AMI_ID" \
    --instance-type "$SELECTED_INSTANCE_TYPE" \
    --key-name "$KEY_NAME" \
    --security-group-ids "$CLIENT_SG_ID" \
    --subnet-id "$SELECTED_SUBNET_ID" \
    --iam-instance-profile "Name=$INSTANCE_PROFILE_NAME" \
    --tag-specifications
 "ResourceType=instance,Tags=[{Key=Name,Value=MSKTutorialClient-
${RANDOM_SUFFIX}},{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-
managed-streaming-for-apache-kafka-gs}]" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to launch EC2 instance: $INSTANCE_RESPONSE"
fi

# Extract the instance ID using grep
```

```
INSTANCE_ID=$(echo "$INSTANCE_RESPONSE" | grep -o '"InstanceId": "[^"]*' | head
 -1 | cut -d'"' -f4)

if [ -z "$INSTANCE_ID" ]; then
    handle_error "Failed to extract instance ID from response:
 $INSTANCE_RESPONSE"
fi

echo "EC2 instance launched successfully. ID: $INSTANCE_ID"
echo "Waiting for instance to be running..."

# Wait for the instance to be running
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

if [ $? -ne 0 ]; then
    handle_error "Instance failed to reach running state"
fi

# Wait a bit more for the instance to initialize
echo "Instance is running. Waiting 30 seconds for initialization..."
sleep 30

# Get public DNS name of instance
CLIENT_DNS=$(aws ec2 describe-instances \
    --instance-ids "$INSTANCE_ID" \
    --query 'Reservations[0].Instances[0].PublicDnsName' \
    --output text)

if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
    echo "Warning: Could not get public DNS name for instance. Trying public
 IP..."
    CLIENT_DNS=$(aws ec2 describe-instances \
        --instance-ids "$INSTANCE_ID" \
        --query 'Reservations[0].Instances[0].PublicIpAddress' \
        --output text)

    if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
        handle_error "Failed to get public DNS name or IP address for instance"
    fi
fi

echo "Client instance DNS/IP: $CLIENT_DNS"
echo "============================================="
# Get bootstrap brokers with improved logic
```

```
echo "Getting bootstrap brokers"
MAX_RETRIES=10
RETRY_COUNT=0
BOOTSTRAP_BROKERS=""
AUTH_METHOD=""

while [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; do
    # Get the full bootstrap brokers response
    BOOTSTRAP_RESPONSE=$(aws kafka get-bootstrap-brokers \
        --cluster-arn "$CLUSTER_ARN" 2>/dev/null)

    if [ $? -eq 0 ] && [ -n "$BOOTSTRAP_RESPONSE" ]; then
        # Try to get IAM authentication brokers first using grep
        BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringSaslIam": "[^"]*' | cut -d'"' -f4)
        if [ -n "$BOOTSTRAP_BROKERS" ]; then
            AUTH_METHOD="IAM"
        else
            # Fall back to TLS authentication
            BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringTls": "[^"]*' | cut -d'"' -f4)
            if [ -n "$BOOTSTRAP_BROKERS" ]; then
                AUTH_METHOD="TLS"
            fi
        fi
    fi

    RETRY_COUNT=$((RETRY_COUNT + 1))

    if [ "$RETRY_COUNT" -ge "$MAX_RETRIES" ]; then
        echo "Warning: Could not get bootstrap brokers after $MAX_RETRIES
 attempts."
        echo "You may need to manually retrieve them later using:"
        echo "aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
        BOOTSTRAP_BROKERS="BOOTSTRAP_BROKERS_NOT_AVAILABLE"
        AUTH_METHOD="UNKNOWN"
        break
    fi

    if [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; then
        echo "Bootstrap brokers not available yet. Retrying in 30 seconds...
 (Attempt $RETRY_COUNT/$MAX_RETRIES)"
        sleep 30
    fi
```

```
done

echo "Bootstrap brokers: $BOOTSTRAP_BROKERS"
echo "Authentication method: $AUTH_METHOD"
echo "============================================="

# Create setup script for the client machine
echo "Creating setup script for the client machine"
cat > setup_client.sh << 'EOF'
#!/bin/bash

# Set up logging
LOG_FILE="client_setup_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting client setup"
echo "============================================="

# Install Java
echo "Installing Java"
sudo yum -y install java-11

# Set environment variables
echo "Setting up environment variables"
export KAFKA_VERSION="3.6.0"
echo "KAFKA_VERSION=$KAFKA_VERSION"

# Download and extract Apache Kafka
echo "Downloading Apache Kafka"
wget https://archive.apache.org/dist/kafka/$KAFKA_VERSION/kafka_2.13-
$KAFKA_VERSION.tgz
if [ $? -ne 0 ]; then
    echo "Failed to download Kafka. Trying alternative mirror..."
    wget https://www.apache.org/dyn/closer.cgi?path=/kafka/$KAFKA_VERSION/
kafka_2.13-$KAFKA_VERSION.tgz
fi

echo "Extracting Kafka"
tar -xzf kafka_2.13-$KAFKA_VERSION.tgz
export KAFKA_ROOT=$(pwd)/kafka_2.13-$KAFKA_VERSION
echo "KAFKA_ROOT=$KAFKA_ROOT"

# Download the MSK IAM authentication package (needed for both IAM and TLS)
echo "Downloading MSK IAM authentication package"
```

```
cd $KAFKA_ROOT/libs
wget https://github.com/aws/aws-msk-iam-auth/releases/latest/download/aws-msk-
iam-auth-1.1.6-all.jar
if [ $? -ne 0 ]; then
    echo "Failed to download specific version. Trying to get latest version..."
    LATEST_VERSION=$(curl -s https://api.github.com/repos/aws/aws-msk-iam-auth/
releases/latest | grep -o '"tag_name": "[^"]*' | cut -d'"' -f4)
    wget https://github.com/aws/aws-msk-iam-auth/releases/download/
$LATEST_VERSION/aws-msk-iam-auth-$LATEST_VERSION-all.jar
    if [ $? -ne 0 ]; then
        echo "Failed to download IAM auth package. Please check the URL and try
 again."
        exit 1
    fi
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-$LATEST_VERSION-all.jar
else
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
fi
echo "CLASSPATH=$CLASSPATH"

# Create client properties file based on authentication method
echo "Creating client properties file"
cd $KAFKA_ROOT/config

# The AUTH_METHOD_PLACEHOLDER will be replaced by the script
AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"

if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "Configuring for IAM authentication"
    cat > client.properties << 'EOT'
security.protocol=SASL_SSL
sasl.mechanism=AWS_MSK_IAM
sasl.jaas.config=software.amazon.msk.auth.iam.IAMLoginModule required;
sasl.client.callback.handler.class=software.amazon.msk.auth.iam.IAMClientCallbackHandler
EOT
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "Configuring for TLS authentication"
    cat > client.properties << 'EOT'
security.protocol=SSL
EOT
else
    echo "Unknown authentication method. Creating basic TLS configuration."
    cat > client.properties << 'EOT'
security.protocol=SSL
```

```
EOT
fi

echo "Client setup completed"
echo "============================================="

# Create a script to set up environment variables
cat > ~/setup_env.sh << 'EOT'
#!/bin/bash
export KAFKA_VERSION="3.6.0"
export KAFKA_ROOT=~/kafka_2.13-$KAFKA_VERSION
export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
export BOOTSTRAP_SERVER="BOOTSTRAP_SERVER_PLACEHOLDER"
export AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"

echo "Environment variables set:"
echo "KAFKA_VERSION=$KAFKA_VERSION"
echo "KAFKA_ROOT=$KAFKA_ROOT"
echo "CLASSPATH=$CLASSPATH"
echo "BOOTSTRAP_SERVER=$BOOTSTRAP_SERVER"
echo "AUTH_METHOD=$AUTH_METHOD"
EOT

chmod +x ~/setup_env.sh

echo "Created environment setup script: ~/setup_env.sh"
echo "Run 'source ~/setup_env.sh' to set up your environment"
EOF

# Replace placeholders in the setup script
if [ -n "$BOOTSTRAP_BROKERS" ] && [ "$BOOTSTRAP_BROKERS" != "None" ] &&
 [ "$BOOTSTRAP_BROKERS" != "BOOTSTRAP_BROKERS_NOT_AVAILABLE" ]; then
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|$BOOTSTRAP_BROKERS|g" setup_client.sh
else
    # If bootstrap brokers are not available, provide instructions to get them
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|\$(aws kafka get-bootstrap-brokers
 --cluster-arn $CLUSTER_ARN --query 'BootstrapBrokerStringTls' --output text)|g"
 setup_client.sh
fi

# Replace auth method placeholder
sed -i "s|AUTH_METHOD_PLACEHOLDER|$AUTH_METHOD|g" setup_client.sh

echo "Setup script created"
```

```
echo "============================================="

# Display summary of created resources
echo ""
echo "============================================="
echo "RESOURCE SUMMARY"
echo "============================================="
echo "MSK Cluster ARN: $CLUSTER_ARN"
echo "MSK Cluster Name: $CLUSTER_NAME"
echo "Authentication Method: $AUTH_METHOD"
echo "IAM Policy ARN: $POLICY_ARN"
echo "IAM Role Name: $ROLE_NAME"
echo "IAM Instance Profile: $INSTANCE_PROFILE_NAME"
echo "Client Security Group: $CLIENT_SG_ID"
echo "EC2 Instance ID: $INSTANCE_ID"
echo "EC2 Instance Type: $SELECTED_INSTANCE_TYPE"
echo "EC2 Instance DNS: $CLIENT_DNS"
echo "Key Pair: $KEY_NAME (saved to $KEY_FILE)"
echo "Bootstrap Brokers: $BOOTSTRAP_BROKERS"
echo "============================================="
echo ""

# Instructions for connecting to the instance and setting up the client
echo "NEXT STEPS:"
echo "1. Connect to your EC2 instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS"
echo ""
echo "2. Upload the setup script to your instance:"
echo "   scp -i $KEY_FILE setup_client.sh ec2-user@$CLIENT_DNS:~/"
echo ""
echo "3. Run the setup script on your instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS 'chmod +x ~/setup_client.sh && ~/
setup_client.sh'"
echo ""
echo "4. Source the environment setup script:"
echo "   source ~/setup_env.sh"
echo ""

# Provide different instructions based on authentication method
if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "5. Create a Kafka topic (using IAM authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
```

```
    echo "     --replication-factor 3 \\"
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "5. Create a Kafka topic (using TLS authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --replication-factor 3 \\"
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
else
    echo "5. Manually retrieve bootstrap brokers and configure authentication:"
    echo "   aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
fi

echo ""
echo "8. Verify the topic was created:"
```

```
echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --list \\"
echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
echo "     --command-config \$KAFKA_ROOT/config/client.properties"
echo "============================================="
echo ""

# Ask if user wants to clean up resources
echo ""
echo "=========================================="
echo "CLEANUP CONFIRMATION"
echo "=========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ $CLEANUP_CHOICE =~ ^[Yy]$ ]]; then
    cleanup_resources
    echo "All resources have been cleaned up."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run this script again and choose 'y' when
 prompted."
fi

echo "Script completed successfully!"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateInstanceProfile
  - CreateKeyPair
  - CreatePolicy
  - CreateRole
  - CreateSecurityGroup
  - DeleteCluster

- DeleteInstanceProfile
- DeleteKeyPair
- DeletePolicy
- DeleteRole
- DeleteSecurityGroup
- DescribeAvailabilityZones
- DescribeCluster
- DescribeImages
- DescribeInstanceTypeOfferings
- DescribeInstances
- DescribeKeyPairs
- DescribeSecurityGroups
- DescribeSubnets
- DescribeVpcs
- DetachRolePolicy
- GetBootstrapBrokers
- GetCallerIdentity
- GetInstanceProfile
- GetPolicy
- GetRole
- RemoveRoleFromInstanceProfile
- RevokeSecurityGroupEgress
- RevokeSecurityGroupIngress
- RunInstances
- TerminateInstances
- Attente

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Commencer à utiliser les notifications push

L'exemple de code suivant illustre comment :

- Création d'une application
- Activer les canaux de notification push
- Envoyer une notification push
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# AWS End User Messaging Push Getting Started Script
# This script creates an AWS End User Messaging Push application and demonstrates
# how to enable push notification channels and send a test message.
#
# Prerequisites:
# - AWS CLI installed and configured
# - Appropriate IAM permissions for Pinpoint operations
#
# Usage: ./aws-end-user-messaging-gs.sh [--auto-cleanup]

set -uo pipefail

# Security: Set secure umask for created files
umask 0077

# Set up logging
LOG_DIR="./aws-eump-logs"
mkdir -p "$LOG_DIR"
LOG_FILE="$LOG_DIR/aws-end-user-messaging-push-script-$(date +%Y%m%d-%H%M%S).log"
```

```
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS End User Messaging Push setup script..."
echo "Logging to $LOG_FILE"
echo "Timestamp: $(date)"

# Security: Track created resources for cleanup
declare -a TEMP_FILES=()
declare -a AWS_RESOURCES=()

# Cleanup function with improved security
cleanup() {
    local exit_code=$?
    echo "Cleaning up temporary resources..."

    # Remove temporary files securely
    for temp_file in "${TEMP_FILES[@]+"${TEMP_FILES[@]}"}"; do
        if [ -f "$temp_file" ]; then
            shred -vfz -n 3 "$temp_file" 2>/dev/null || rm -f "$temp_file"
        fi
    done

    # Optionally delete AWS resources
    if [ "${DELETE_AWS_RESOURCES:-false}" = "true" ]; then
        for resource in "${AWS_RESOURCES[@]+"${AWS_RESOURCES[@]}"}"; do
            echo "Deleting AWS resource: $resource"
            aws pinpoint delete-app --application-id "$resource" 2>/dev/null || \
                echo "Warning: Failed to delete application $resource"
        done
    fi

    exit "$exit_code"
}

trap cleanup EXIT INT TERM

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2
    local ignore_error=${3:-false}

    if echo "$output" | grep -qi "error\|exception\|fail"; then
        echo "ERROR: Command failed: $cmd" >&2
```

```
        echo "Error details: $output" >&2

        if [ "$ignore_error" = "true" ]; then
            echo "Ignoring error and continuing..." >&2
            return 1
        else
            return 2
        fi
    fi

    return 0
}

# Function to validate AWS CLI is configured
validate_aws_cli() {
    echo "Validating AWS CLI configuration..."

    # Check if AWS CLI is installed
    if ! command -v aws &> /dev/null; then
        echo "ERROR: AWS CLI is not installed. Please install it first." >&2
        echo "Visit: https://docs.aws.amazon.com/cli/latest/userguide/getting-
started-install.html" >&2
        return 1
    fi

    # Check AWS CLI version
    AWS_VERSION=$(aws --version 2>&1 | head -n1)
    echo "AWS CLI version: $AWS_VERSION"

    # Verify credentials are set (check for credential env vars or config file)
    if ! aws sts get-caller-identity &> /dev/null; then
        echo "ERROR: AWS CLI credentials are not configured or invalid." >&2
        echo "Please configure credentials via environment variables, credential
 file, or 'aws configure'" >&2
        return 1
    fi

    # Get current AWS identity and region
    CALLER_IDENTITY=$(aws sts get-caller-identity)
    CURRENT_REGION=$(aws configure get region 2>/dev/null || echo "us-east-1")
    echo "AWS CLI configured for:"
    echo "$CALLER_IDENTITY"
    echo "Current region: $CURRENT_REGION"
    echo ""
```

```
    return 0
}

# Function to check if jq is available for JSON parsing
check_json_tools() {
    if command -v jq &> /dev/null; then
        USE_JQ=true
        echo "jq is available for JSON parsing"
    else
        USE_JQ=false
        echo "jq is not available, using grep for JSON parsing"
        echo "Consider installing jq for better JSON handling: https://
stedolan.github.io/jq/"
    fi
}

# Function to extract JSON values safely
extract_json_value() {
    local json=$1
    local key=$2

    if [ "$USE_JQ" = "true" ]; then
        echo "$json" | jq -r ".ApplicationResponse.$key // empty" 2>/dev/null ||
 echo ""
    else
        # Fallback to grep method with better validation
        echo "$json" | grep -o "\"$key\": \"[^\"]*" | cut -d'"' -f4 | head -n1 ||
 echo ""
    fi
}

# Function to validate required IAM permissions
validate_permissions() {
    echo "Validating IAM permissions..."

    # Test basic Pinpoint permissions
    if ! aws pinpoint get-apps > /dev/null 2>&1; then
        echo "WARNING: Unable to list Pinpoint applications." >&2
        echo "Please ensure you have appropriate IAM permissions for Pinpoint
 operations." >&2
        echo "Required permissions:" >&2
        echo "  - mobiletargeting:GetApps" >&2
        echo "  - mobiletargeting:CreateApp" >&2
```

```
        echo "  - mobiletargeting:DeleteApp" >&2
        echo "  - mobiletargeting:UpdateGcmChannel" >&2
        echo "  - mobiletargeting:UpdateApnsChannel" >&2
        echo "  - mobiletargeting:SendMessages" >&2
    else
        echo "Basic Pinpoint permissions validated."
    fi
}

# Function to validate input parameters
validate_input() {
    local app_name=$1

    # Validate app name length and characters
    if [ ${#app_name} -gt 64 ]; then
        echo "ERROR: Application name exceeds maximum length of 64 characters"
 >&2
        return 1
    fi

    if ! [[ "$app_name" =~ ^[a-zA-Z0-9_-]+$ ]]; then
        echo "ERROR: Application name contains invalid characters" >&2
        return 1
    fi

    return 0
}

# Function to create secure temporary files
create_temp_file() {
    local temp_file
    temp_file=$(mktemp) || {
        echo "ERROR: Failed to create temporary file" >&2
        return 1
    }
    chmod 600 "$temp_file"
    TEMP_FILES+=("$temp_file")
    echo "$temp_file"
}

# Validate prerequisites
if ! validate_aws_cli; then
    exit 1
fi
```

```
check_json_tools
validate_permissions

# Generate a random suffix for resource names to avoid conflicts
RANDOM_SUFFIX=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 8 | head -n1)
APP_NAME="PushNotificationApp-${RANDOM_SUFFIX}"

# Validate input
if ! validate_input "$APP_NAME"; then
    exit 1
fi

echo "Creating application with name: $APP_NAME"

# Step 1: Create an application
echo "Executing: aws pinpoint create-app --create-application-request Name=
${APP_NAME}"
CREATE_APP_OUTPUT=$(aws pinpoint create-app --create-application-request "Name=
${APP_NAME},tags={project=doc-smith,tutorial=aws-end-user-messaging-gs}" 2>&1)

if ! check_error "$CREATE_APP_OUTPUT" "create-app"; then
    exit 1
fi

echo "Application created successfully:"
echo "$CREATE_APP_OUTPUT"

# Extract the application ID from the output
APP_ID=$(extract_json_value "$CREATE_APP_OUTPUT" "Id")

if [ -z "$APP_ID" ] || [ "$APP_ID" = "null" ]; then
    echo "ERROR: Failed to extract application ID from output" >&2
    echo "Output was: $CREATE_APP_OUTPUT" >&2
    exit 1
fi

echo "Application ID: $APP_ID"
AWS_RESOURCES+=("$APP_ID")

# Step 2: Enable FCM (GCM) channel with a sample API key
echo ""
echo "========================================="
echo "ENABLING FCM (GCM) CHANNEL"
```

```
echo "========================================="
echo "Note: This is using a placeholder API key for demonstration purposes only."
echo "In a production environment, you should use your actual FCM API key from
 Firebase Console."
echo ""
echo "IMPORTANT: The following command will likely fail because we're using a
 placeholder API key."
echo "This is expected behavior for this demonstration script."

echo "Executing: aws pinpoint update-gcm-channel --application-id $APP_ID --gcm-
channel-request ..."
UPDATE_GCM_OUTPUT=$(aws pinpoint update-gcm-channel \
    --application-id "$APP_ID" \
    --gcm-channel-request '{"Enabled": true, "ApiKey": "sample-fcm-api-key-for-
demo-only"}' 2>&1)

# We'll ignore this specific error since we're using a placeholder API key
if check_error "$UPDATE_GCM_OUTPUT" "update-gcm-channel" "true"; then
    echo "FCM channel enabled successfully:"
    echo "$UPDATE_GCM_OUTPUT"
else
    echo "As expected, FCM channel update failed with the placeholder API key."
    echo "Error details: $UPDATE_GCM_OUTPUT"
    echo ""
    echo "To enable FCM in production:"
    echo "1. Go to Firebase Console (https://console.firebase.google.com/)"
    echo "2. Create or select your project"
    echo "3. Go to Project Settings > Cloud Messaging"
    echo "4. Copy the Server Key"
    echo "5. Replace 'sample-fcm-api-key-for-demo-only' with your actual Server
 Key"
fi

# Step 3: Try to enable APNS channel (this will also fail without real
 certificates)
echo ""
echo "========================================="
echo "ENABLING APNS CHANNEL (OPTIONAL)"
echo "========================================="
echo "Attempting to enable APNS channel with placeholder certificate..."
echo "This will also fail without real APNS certificates, which is expected."

# Create a placeholder APNS configuration
```

```
echo "Executing: aws pinpoint update-apns-channel --application-id $APP_ID --
apns-channel-request ..."
UPDATE_APNS_OUTPUT=$(aws pinpoint update-apns-channel \
    --application-id "$APP_ID" \
    --apns-channel-request '{"Enabled": true, "Certificate": "placeholder-
certificate", "PrivateKey": "placeholder-private-key"}' 2>&1)

if check_error "$UPDATE_APNS_OUTPUT" "update-apns-channel" "true"; then
    echo "APNS channel enabled successfully:"
    echo "$UPDATE_APNS_OUTPUT"
else
    echo "As expected, APNS channel update failed with placeholder certificates."
    echo "Error details: $UPDATE_APNS_OUTPUT"
    echo ""
    echo "To enable APNS in production:"
    echo "1. Generate APNS certificates from Apple Developer Console"
    echo "2. Convert certificates to PEM format"
    echo "3. Use the actual certificate and private key in the update-apns-
channel command"
fi

# Step 4: Create message files for different platforms
echo ""
echo "=========================================="
echo "CREATING MESSAGE FILES"
echo "=========================================="

# Create FCM message file securely
GCM_MESSAGE_FILE=$(create_temp_file)
echo "Creating FCM message file..."
cat > "$GCM_MESSAGE_FILE" << 'EOF'
{
  "Addresses": {
    "SAMPLE-DEVICE-TOKEN-FCM": {
      "ChannelType": "GCM"
    }
  },
  "MessageConfiguration": {
    "GCMMessage": {
      "Action": "OPEN_APP",
      "Body": "Hello from AWS End User Messaging Push! This is an FCM
 notification.",
      "Priority": "normal",
      "SilentPush": false,
```

```
      "Title": "My First FCM Push Notification",
      "TimeToLive": 30,
      "Data": {
        "key1": "value1",
        "key2": "value2"
      }
    }
  }
}
EOF

# Create APNS message file securely
APNS_MESSAGE_FILE=$(create_temp_file)
echo "Creating APNS message file..."
cat > "$APNS_MESSAGE_FILE" << 'EOF'
{
  "Addresses": {
    "SAMPLE-DEVICE-TOKEN-APNS": {
      "ChannelType": "APNS"
    }
  },
  "MessageConfiguration": {
    "APNSMessage": {
      "Action": "OPEN_APP",
      "Body": "Hello from AWS End User Messaging Push! This is an APNS
 notification.",
      "Priority": "normal",
      "SilentPush": false,
      "Title": "My First APNS Push Notification",
      "TimeToLive": 30,
      "Badge": 1,
      "Sound": "default"
    }
  }
}
EOF

echo "Message files created:"
echo "- FCM message file (for FCM/Android)"
echo "- APNS message file (for APNS/iOS)"
echo ""
echo "Note: These messages use placeholder device tokens and will not actually be
 delivered."
```

```
echo "To send real messages, you would need to replace the sample device tokens
 with actual ones."

# Step 5: Demonstrate how to send messages (this will fail with placeholder
 tokens)
echo ""
echo "========================================="
echo "DEMONSTRATING MESSAGE SENDING"
echo "========================================="
echo "Attempting to send FCM message (will fail with placeholder token)..."

echo "Executing: aws pinpoint send-messages --application-id $APP_ID --message-
request file://<gcm-message>"
SEND_FCM_OUTPUT=$(aws pinpoint send-messages \
    --application-id "$APP_ID" \
    --message-request "file://$GCM_MESSAGE_FILE" 2>&1)

if check_error "$SEND_FCM_OUTPUT" "send-messages (FCM)" "true"; then
    echo "FCM message sent successfully:"
    echo "$SEND_FCM_OUTPUT"
else
    echo "As expected, FCM message sending failed with placeholder token."
    echo "Error details: $SEND_FCM_OUTPUT"
fi

echo ""
echo "Attempting to send APNS message (will fail with placeholder token)..."

echo "Executing: aws pinpoint send-messages --application-id $APP_ID --message-
request file://<apns-message>"
SEND_APNS_OUTPUT=$(aws pinpoint send-messages \
    --application-id "$APP_ID" \
    --message-request "file://$APNS_MESSAGE_FILE" 2>&1)

if check_error "$SEND_APNS_OUTPUT" "send-messages (APNS)" "true"; then
    echo "APNS message sent successfully:"
    echo "$SEND_APNS_OUTPUT"
else
    echo "As expected, APNS message sending failed with placeholder token."
    echo "Error details: $SEND_APNS_OUTPUT"
fi

# Step 6: Show application details
echo ""
```

```
echo "============================================"
echo "APPLICATION DETAILS"
echo "============================================"
echo "Retrieving application details..."

echo "Executing: aws pinpoint get-app --application-id $APP_ID"
GET_APP_OUTPUT=$(aws pinpoint get-app --application-id "$APP_ID" 2>&1)
if check_error "$GET_APP_OUTPUT" "get-app"; then
    echo "Application details:"
    echo "$GET_APP_OUTPUT"
fi

# Display summary of created resources
echo ""
echo "============================================"
echo "RESOURCES CREATED"
echo "============================================"
echo "AWS Resources:"
for resource in "${AWS_RESOURCES[@]+"${AWS_RESOURCES[@]}"}"; do
    echo "- Application: $resource"
done

echo ""
echo "Files created:"
echo "- $LOG_FILE (script log)"

# Auto-cleanup information
echo ""
echo "============================================"
echo "CLEANUP INFORMATION"
echo "============================================"
echo "This script created AWS resources that may incur charges."
echo "AWS resources will be automatically cleaned up on script exit."
echo ""
echo "To manually delete resources later, use:"
echo "  aws pinpoint delete-app --application-id $APP_ID"

# Set flag to delete AWS resources on cleanup
DELETE_AWS_RESOURCES=true

echo ""
echo "============================================"
echo "SCRIPT COMPLETED SUCCESSFULLY"
echo "============================================"
```

```
echo "This script demonstrated:"
echo "1. Creating an AWS End User Messaging Push application"
echo "2. Attempting to enable FCM and APNS channels (with placeholder
 credentials)"
echo "3. Creating message templates for different platforms"
echo "4. Demonstrating message sending commands (with placeholder tokens)"
echo "5. Retrieving application details"
echo "6. Proper cleanup of resources"
echo ""
echo "Security best practices implemented:"
echo "- Secure temporary file handling with restricted permissions"
echo "- Input validation for application names"
echo "- Error handling with proper exit codes"
echo "- Credential validation before AWS operations"
echo "- Automatic cleanup on script exit"
echo "- Secure file destruction for sensitive data"
echo ""
echo "For production use:"
echo "- Replace placeholder API keys with real FCM server keys"
echo "- Replace placeholder certificates with real APNS certificates"
echo "- Replace placeholder device tokens with real device tokens"
echo "- Use AWS IAM roles instead of long-term credentials"
echo "- Implement comprehensive error handling"
echo "- Store sensitive credentials in AWS Secrets Manager or Parameter Store"
echo "- Enable CloudTrail for audit logging"
echo "- Use VPC endpoints for private AWS API access"
echo ""
echo "Log file: $LOG_FILE"
echo "Script completed at: $(date)"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - CreateApp
  - DeleteApp
  - GetApp
  - GetApps
  - GetCallerIdentity
  - SendMessages
  - UpdateApnsChannel
  - UpdateGcmChannel

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Commencer à utiliser les moteurs de recherche et d'analyse

L'exemple de code suivant illustre comment :

- Création d'un domaine OpenSearch de service
- Téléchargez des données sur votre domaine
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Amazon OpenSearch Service Getting Started Script - Version 8 Fixed
# This script creates an OpenSearch domain, uploads data, searches documents, and
 cleans up resources
# Based on the tested and working 4-tutorial-final.md

# SECURITY IMPROVEMENTS IN THIS VERSION:
# 1. Removed hardcoded passwords - now generated securely
# 2. Improved access policy to use principle of least privilege
# 3. Added credential masking in logs
# 4. Removed credentials from command output
# 5. Added input validation for AWS region
# 6. Improved error handling and validation
# 7. Added secure temporary file handling
# 8. Removed unnecessary curl insecurity flags
# 9. Added better secret management practices
# 10. Improved resource tagging for better governance
```

```
set -e  # Exit on any error

# Set up logging with restricted permissions
LOG_FILE="opensearch_tutorial_v8_fixed.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon OpenSearch Service tutorial script v8-fixed at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Track if domain was successfully created
DOMAIN_CREATED=false
DOMAIN_ACTIVE=false

# Secure temporary directory
TEMP_DIR=$(mktemp -d)
trap 'rm -rf "$TEMP_DIR"' EXIT
chmod 700 "$TEMP_DIR"

# Error handling function
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    if [[ "$DOMAIN_CREATED" == "true" ]]; then
        echo "Checking if domain $DOMAIN_NAME exists before attempting to
 delete..."

        # Check if domain exists before trying to delete
        if aws opensearch describe-domain --domain-name "$DOMAIN_NAME" > /dev/
null 2>&1; then
            echo "Domain $DOMAIN_NAME exists. Proceeding with deletion."
            aws opensearch delete-domain --domain-name "$DOMAIN_NAME"
            echo "Domain deletion initiated. This may take several minutes to
 complete."
```

```
        else
            echo "Domain $DOMAIN_NAME does not exist or is not accessible. No
 deletion needed."
        fi
    else
        echo "No domain was successfully created. Nothing to clean up."
    fi

    # Securely clean up sensitive files
    if [[ -d "$TEMP_DIR" ]]; then
        shred -vfz -n 3 "$TEMP_DIR"/* 2>/dev/null || true
        rm -rf "$TEMP_DIR"
    fi
}

# Generate a random identifier for resource names to avoid conflicts
RANDOM_ID=$(openssl rand -hex 4)
DOMAIN_NAME="movies-${RANDOM_ID}"
MASTER_USER="master-user"

# Generate secure password using openssl instead of hardcoding
MASTER_PASSWORD=$(openssl rand -base64 32 | tr -d "=+/" | cut -c1-25)

# Mask password in logs
echo "Using domain name: $DOMAIN_NAME"
echo "Using master username: $MASTER_USER"
echo "Using master password: [REDACTED]"

# Get AWS account ID (matches tutorial)
echo "Retrieving AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ $? -ne 0 ]] || [[ -z "$ACCOUNT_ID" ]]; then
    handle_error "Failed to retrieve AWS account ID. Please check your AWS
 credentials."
fi
echo "AWS Account ID: $ACCOUNT_ID"

# Get current region (matches tutorial)
echo "Retrieving current AWS region..."
AWS_REGION=$(aws configure get region)
if [[ -z "$AWS_REGION" ]]; then
    AWS_REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $AWS_REGION"
else
```

```
    echo "Using AWS region: $AWS_REGION"
fi

# Validate AWS region format
if ! [[ "$AWS_REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]$ ]]; then
    handle_error "Invalid AWS region format: $AWS_REGION"
fi

# Step 1: Create an OpenSearch Service Domain
echo "Creating OpenSearch Service domain..."
echo "This may take 15-30 minutes to complete."

# SECURITY IMPROVED: Use least privilege access policy
# This policy restricts access to specific actions and should be further
 restricted to specific principals in production
# Create the domain using CloudFormation (handles the 15-20 min creation wait)
SCRIPT_DIR="$(cd "$(dirname "${BASH_SOURCE[0]}")" && pwd)"
STACK_NAME="tutorial-opensearch-${RANDOM_ID}"

echo "Creating domain $DOMAIN_NAME via CloudFormation..."
echo "This typically takes 15-20 minutes. CloudFormation will wait for
 completion."
aws cloudformation deploy \
  --template-file "$SCRIPT_DIR/cfn-opensearch-domain.yaml" \
  --stack-name "$STACK_NAME" \
  --parameter-overrides \
    DomainName="$DOMAIN_NAME" \
    MasterUserName="$MASTER_USER" \
    MasterUserPassword="$MASTER_PASSWORD" \
  --tags project=doc-smith tutorial=opensearch-service-gs \
  --no-fail-on-empty-changeset 2>&1 || handle_error "CloudFormation stack
 creation failed"

DOMAIN_CREATED=true
echo "Domain created successfully via CloudFormation."

# Get domain endpoint from stack outputs
DOMAIN_ENDPOINT=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query 'Stacks[0].Outputs[?OutputKey==`DomainEndpoint`].OutputValue' \
  --output text)

if [[ -z "$DOMAIN_ENDPOINT" ]] || [[ "$DOMAIN_ENDPOINT" == "None" ]]; then
    handle_error "Failed to get domain endpoint from CloudFormation outputs"
```

```
fi

echo "Domain endpoint: $DOMAIN_ENDPOINT"

# Wait for fine-grained access control to be fully ready
echo "Waiting for fine-grained access control to initialize..."
sleep 120

# Verify variables are set correctly (matches tutorial)
echo "Verifying configuration..."
echo "Domain endpoint: $DOMAIN_ENDPOINT"
echo "Master user: $MASTER_USER"
echo "Password set: $(if [ -n "$MASTER_PASSWORD" ]; then echo "Yes"; else echo
 "No"; fi)"

# Step 2: Upload Data to the Domain
echo "Preparing to upload data to the domain..."

# Create a file for the single document (matches tutorial exactly)
echo "Creating single document JSON file..."
SINGLE_MOVIE_FILE="$TEMP_DIR/single_movie.json"
cat > "$SINGLE_MOVIE_FILE" << 'EOF'
{
  "director": "Burton, Tim",
  "genre": ["Comedy","Sci-Fi"],
  "year": 1996,
  "actor": ["Jack Nicholson","Pierce Brosnan","Sarah Jessica Parker"],
  "title": "Mars Attacks!"
}
EOF
chmod 600 "$SINGLE_MOVIE_FILE"

# Create a file for bulk documents (matches tutorial exactly)
echo "Creating bulk documents JSON file..."
BULK_MOVIES_FILE="$TEMP_DIR/bulk_movies.json"
cat > "$BULK_MOVIES_FILE" << 'EOF'
{ "index" : { "_index": "movies", "_id" : "2" } }
{"director": "Frankenheimer, John", "genre": ["Drama", "Mystery", "Thriller",
 "Crime"], "year": 1962, "actor": ["Lansbury, Angela", "Sinatra, Frank", "Leigh,
 Janet", "Harvey, Laurence", "Silva, Henry", "Frees, Paul", "Gregory, James",
 "Bissell, Whit", "McGiver, John", "Parrish, Leslie", "Edwards, James", "Flowers,
 Bess", "Dhiegh, Khigh", "Payne, Julie", "Kleeb, Helen", "Gray, Joe", "Nalder,
 Reggie", "Stevens, Bert", "Masters, Michael", "Lowell, Tom"], "title": "The
 Manchurian Candidate"}
```

```
{ "index" : { "_index": "movies", "_id" : "3" } }
{"director": "Baird, Stuart", "genre": ["Action", "Crime", "Thriller"], "year":
 1998, "actor": ["Downey Jr., Robert", "Jones, Tommy Lee", "Snipes, Wesley",
 "Pantoliano, Joe", "Jacob, Irène", "Nelligan, Kate", "Roebuck, Daniel",
 "Malahide, Patrick", "Richardson, LaTanya", "Wood, Tom", "Kosik, Thomas",
 "Stellate, Nick", "Minkoff, Robert", "Brown, Spitfire", "Foster, Reese",
 "Spielbauer, Bruce", "Mukherji, Kevin", "Cray, Ed", "Fordham, David", "Jett,
 Charlie"], "title": "U.S. Marshals"}
{ "index" : { "_index": "movies", "_id" : "4" } }
{"director": "Ray, Nicholas", "genre": ["Drama", "Romance"], "year": 1955,
 "actor": ["Hopper, Dennis", "Wood, Natalie", "Dean, James", "Mineo, Sal",
 "Backus, Jim", "Platt, Edward", "Ray, Nicholas", "Hopper, William", "Allen,
 Corey", "Birch, Paul", "Hudson, Rochelle", "Doran, Ann", "Hicks, Chuck", "Leigh,
 Nelson", "Williams, Robert", "Wessel, Dick", "Bryar, Paul", "Sessions, Almira",
 "McMahon, David", "Peters Jr., House"], "title": "Rebel Without a Cause"}
EOF
chmod 600 "$BULK_MOVIES_FILE"

# Check if curl is installed
if ! command -v curl &> /dev/null; then
    echo "Warning: curl is not installed. Skipping data upload and search steps."
    echo "You can manually upload the data later using the commands in the
 tutorial."
else
    # IMPROVED: Test authentication with multiple approaches
    echo "Testing authentication with the OpenSearch domain..."
    echo "This test checks if fine-grained access control is ready for data
 operations."

    # Create credentials file for secure handling
    CREDENTIALS_FILE="$TEMP_DIR/.credentials"
    echo "$MASTER_USER:$MASTER_PASSWORD" > "$CREDENTIALS_FILE"
    chmod 600 "$CREDENTIALS_FILE"

    # Test 1: Basic authentication with root endpoint
    echo "Testing basic authentication with root endpoint..."
    AUTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
        --netrc-file "$CREDENTIALS_FILE" \
        --request GET \
        "https://${DOMAIN_ENDPOINT}/" 2>&1)

    echo "Basic auth test result:"
    echo "$AUTH_TEST_RESULT"
```

```
    # Extract HTTP status code
    HTTP_CODE=$(echo "$AUTH_TEST_RESULT" | grep "HTTP_CODE:" | cut -d: -f2)

    # Function to check if HTTP code is 2xx
    is_success_code() {
        local code=$1
        if [[ "$code" =~ ^2[0-9][0-9]$ ]]; then
            return 0
        else
            return 1
        fi
    }

    # Check if basic authentication test was successful (200 or 2xx status codes)
    if is_success_code "$HTTP_CODE"; then
        echo "# Basic authentication test successful! (HTTP $HTTP_CODE)"
        AUTH_SUCCESS=true
        AUTH_METHOD="basic"
    else
        echo "Basic authentication failed with HTTP code: $HTTP_CODE"

        # Test 2: Try cluster health endpoint
        echo "Testing with cluster health endpoint..."
        HEALTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request GET \
            "https://${DOMAIN_ENDPOINT}/_cluster/health" 2>&1)

        echo "Cluster health test result:"
        echo "$HEALTH_TEST_RESULT"

        HEALTH_HTTP_CODE=$(echo "$HEALTH_TEST_RESULT" | grep "HTTP_CODE:" | cut -
d: -f2)

        if is_success_code "$HEALTH_HTTP_CODE"; then
            echo "# Cluster health authentication test successful! (HTTP
 $HEALTH_HTTP_CODE)"
            AUTH_SUCCESS=true
            AUTH_METHOD="basic"
        else
            echo "Cluster health authentication also failed with HTTP code:
 $HEALTH_HTTP_CODE"

            # Check for specific error patterns
```

```
            if echo "$AUTH_TEST_RESULT" | grep -q "anonymous is not authorized";
 then
                echo "Error: Request is being treated as anonymous
 (authentication not working)"
            elif echo "$AUTH_TEST_RESULT" | grep -q "Unauthorized"; then
                echo "Error: Authentication credentials rejected"
            elif echo "$AUTH_TEST_RESULT" | grep -q "Forbidden"; then
                echo "Error: Authentication succeeded but access is forbidden"
            fi

            echo "Waiting additional time and retrying with exponential
 backoff..."

            # Retry authentication test with exponential backoff
            AUTH_RETRY_COUNT=0
            MAX_AUTH_RETRIES=5
            WAIT_TIME=60
            AUTH_SUCCESS=false

            while [[ $AUTH_RETRY_COUNT -lt $MAX_AUTH_RETRIES ]]; do
                echo "Retrying authentication test (attempt $((AUTH_RETRY_COUNT
+1))/$MAX_AUTH_RETRIES) after ${WAIT_TIME} seconds..."
                sleep $WAIT_TIME

                # Try both endpoints
                AUTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
                    --netrc-file "$CREDENTIALS_FILE" \
                    --request GET \
                    "https://${DOMAIN_ENDPOINT}/" 2>&1)

                HTTP_CODE=$(echo "$AUTH_TEST_RESULT" | grep "HTTP_CODE:" | cut -
d: -f2)

                echo "Retry result (HTTP $HTTP_CODE):"
                echo "$AUTH_TEST_RESULT"

                if is_success_code "$HTTP_CODE"; then
                    echo "# Authentication test successful after retry! (HTTP
 $HTTP_CODE)"
                    AUTH_SUCCESS=true
                    AUTH_METHOD="basic"
                    break
                fi
```

```
                # Also try cluster health
                HEALTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
                    --netrc-file "$CREDENTIALS_FILE" \
                    --request GET \
                    "https://${DOMAIN_ENDPOINT}/_cluster/health" 2>&1)

                HEALTH_HTTP_CODE=$(echo "$HEALTH_TEST_RESULT" | grep "HTTP_CODE:"
| cut -d: -f2)

                if is_success_code "$HEALTH_HTTP_CODE"; then
                    echo "# Cluster health authentication successful after retry!
(HTTP $HEALTH_HTTP_CODE)"
                    AUTH_SUCCESS=true
                    AUTH_METHOD="basic"
                    break
                fi

                AUTH_RETRY_COUNT=$((AUTH_RETRY_COUNT+1))
                # Exponential backoff: double the wait time each retry (max 10
minutes)
                WAIT_TIME=$((WAIT_TIME * 2))
                if [[ $WAIT_TIME -gt 600 ]]; then
                    WAIT_TIME=600
                fi
            done
        fi
    fi

    # Proceed with data operations if authentication is working
    if [[ "$AUTH_SUCCESS" == "true" ]]; then
        echo "Authentication successful using $AUTH_METHOD method. Proceeding
with data operations."

        # Upload single document (matches tutorial exactly)
        echo "Uploading single document..."
        UPLOAD_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request PUT \
            --header 'Content-Type: application/json' \
            --data @"$SINGLE_MOVIE_FILE" \
            "https://${DOMAIN_ENDPOINT}/movies/_doc/1" 2>&1)

        echo "Upload response:"
        echo "$UPLOAD_RESULT"
```

```
        UPLOAD_HTTP_CODE=$(echo "$UPLOAD_RESULT" | grep "HTTP_CODE:" | cut -d: -
f2)
        if is_success_code "$UPLOAD_HTTP_CODE" && echo "$UPLOAD_RESULT" | grep -q
 '"result"'; then
            echo "# Single document uploaded successfully! (HTTP
 $UPLOAD_HTTP_CODE)"
        else
            echo "# Warning: Single document upload may have failed (HTTP
 $UPLOAD_HTTP_CODE)"
        fi

        # Upload bulk documents (matches tutorial exactly)
        echo "Uploading bulk documents..."
        BULK_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request POST \
            --header 'Content-Type: application/x-ndjson' \
            --data-binary @"$BULK_MOVIES_FILE" \
            "https://${DOMAIN_ENDPOINT}/movies/_bulk" 2>&1)

        echo "Bulk upload response:"
        echo "$BULK_RESULT"

        BULK_HTTP_CODE=$(echo "$BULK_RESULT" | grep "HTTP_CODE:" | cut -d: -f2)
        if is_success_code "$BULK_HTTP_CODE" && echo "$BULK_RESULT" | grep -q
 '"errors": false'; then
            echo "# Bulk documents uploaded successfully! (HTTP $BULK_HTTP_CODE)"
        else
            echo "# Warning: Bulk document upload may have failed (HTTP
 $BULK_HTTP_CODE)"
        fi

        # Wait a moment for indexing
        echo "Waiting for documents to be indexed..."
        sleep 5

        # Step 3: Search Documents (matches tutorial exactly)
        echo "Searching for documents containing 'mars'..."
        SEARCH_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request GET \
            "https://${DOMAIN_ENDPOINT}/movies/_search?q=mars&pretty=true" 2>&1)
```

```
        SEARCH_HTTP_CODE=$(echo "$SEARCH_RESULT" | grep "HTTP_CODE:" | cut -d: -
f2)
        echo "Search results for 'mars' (HTTP $SEARCH_HTTP_CODE):"
        echo "$SEARCH_RESULT"

        echo "Searching for documents containing 'rebel'..."
        REBEL_SEARCH=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request GET \
            "https://${DOMAIN_ENDPOINT}/movies/_search?q=rebel&pretty=true" 2>&1)

        REBEL_HTTP_CODE=$(echo "$REBEL_SEARCH" | grep "HTTP_CODE:" | cut -d: -f2)
        echo "Search results for 'rebel' (HTTP $REBEL_HTTP_CODE):"
        echo "$REBEL_SEARCH"

        # Verify search results
        if is_success_code "$SEARCH_HTTP_CODE" && echo "$SEARCH_RESULT" | grep -q
 '"hits"'; then
            echo "# Search functionality is working!"
        else
            echo "# Warning: Search may not be working properly."
        fi

    else
        echo ""
        echo "========================================="
        echo "AUTHENTICATION TROUBLESHOOTING"
        echo "========================================="
        echo "Authentication failed after all retries. This may be due to:"
        echo "1. Fine-grained access control not fully initialized (most common)"
        echo "2. Domain configuration issues"
        echo "3. Network connectivity issues"
        echo "4. AWS credentials or permissions issues"
        echo ""
        echo "DOMAIN CONFIGURATION DEBUG:"
        echo "Let's check the domain configuration..."

        # Debug domain configuration
        DOMAIN_CONFIG=$(aws opensearch describe-domain --domain-
name "$DOMAIN_NAME" --query 'DomainStatus.{AdvancedSecurityOptions:
 AdvancedSecurityOptions, AccessPolicies: AccessPolicies}' --output json 2>&1)
        echo "Domain configuration:"
        echo "$DOMAIN_CONFIG"
```

```
        echo ""
        echo "MANUAL TESTING COMMANDS:"
        echo "You can try these commands manually in 10-15 minutes:"
        echo ""
        echo "# Test basic authentication:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" \"https://
${DOMAIN_ENDPOINT}/\""
        echo ""
        echo "# Test cluster health:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" \"https://
${DOMAIN_ENDPOINT}/_cluster/health\""
        echo ""
        echo "# Upload single document:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" --request PUT --
header 'Content-Type: application/json' --data @single_movie.json \"https://
${DOMAIN_ENDPOINT}/movies/_doc/1\""
        echo ""
        echo "# Search for documents:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" \"https://
${DOMAIN_ENDPOINT}/movies/_search?q=mars&pretty=true\""
        echo ""
        echo "TROUBLESHOOTING TIPS:"
        echo "- Wait 10-15 more minutes and try the manual commands"
        echo "- Check AWS CloudTrail logs for authentication errors"
        echo "- Verify your AWS region is correct: $AWS_REGION"
        echo "- Ensure your AWS credentials have OpenSearch permissions"
        echo "- Try accessing OpenSearch Dashboards to verify the master user
 works"
        echo ""
        echo "Skipping data upload and search operations for now."
        echo "The domain is created and accessible via OpenSearch Dashboards."
    fi

    # Securely remove credentials file
    shred -vfz -n 3 "$CREDENTIALS_FILE" 2>/dev/null || true
fi

# Display OpenSearch Dashboards URL (matches tutorial)
echo ""
echo "============================================"
echo "OPENSEARCH DASHBOARDS ACCESS"
echo "============================================"
echo "OpenSearch Dashboards URL: https://${DOMAIN_ENDPOINT}/_dashboards/"
echo "Username: $MASTER_USER"
```

```
echo "Password: [REDACTED - stored securely]"
echo ""
echo "You can access OpenSearch Dashboards using the credentials provided during
 domain creation."
echo "If you uploaded data successfully, you can create an index pattern for
 'movies'."
echo ""

# Summary of created resources
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
echo "OpenSearch Domain Name: $DOMAIN_NAME"
echo "OpenSearch Domain Endpoint: $DOMAIN_ENDPOINT"
echo "AWS Region: $AWS_REGION"
echo "Master Username: $MASTER_USER"
echo ""
echo "ESTIMATED COST: ~$0.038/hour (~$0.91/day) until deleted"
echo ""
echo "Make sure to save the domain name and endpoint for future reference."
echo ""

# Auto-confirm cleanup: Yes, clean up all resources
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Automatically cleaning up all created resources..."
CLEANUP_CHOICE="y"

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."
    if [ -n "${STACK_NAME:-}" ]; then
        echo "Deleting CloudFormation stack $STACK_NAME..."
        aws cloudformation delete-stack --stack-name "$STACK_NAME"
        echo "# Stack deletion initiated. This may take several minutes."
        echo "  Monitor: aws cloudformation describe-stacks --stack-name
 $STACK_NAME"
    else
        aws opensearch delete-domain --domain-name "$DOMAIN_NAME" 2>/dev/null
        echo "# Domain deletion initiated."
    fi
else
```

```
    echo "Resources will NOT be deleted automatically."
    echo ""
    echo "To delete later:"
    echo "  aws cloudformation delete-stack --stack-name ${STACK_NAME:-tutorial-
opensearch-XXXX}"
    echo ""
    echo "# IMPORTANT: Keeping these resources will incur ongoing AWS charges!"
fi

# Clean up temporary files (handled by trap)
echo "Cleaning up temporary files..."

# Disable the trap since we're handling cleanup manually
trap - EXIT

# Final cleanup
rm -rf "$TEMP_DIR" 2>/dev/null || true

echo ""
echo "============================================="
echo "SCRIPT COMPLETED SUCCESSFULLY"
echo "============================================="
echo "Script completed at $(date)"
echo "All output has been logged to: $LOG_FILE"
echo ""
echo "Next steps:"
echo "1. Access OpenSearch Dashboards at: https://${DOMAIN_ENDPOINT}/
_dashboards/"
echo "2. Create visualizations and dashboards"
echo "3. Explore the OpenSearch API"
echo "4. Remember to delete resources when done to avoid charges"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - CreateDomain
  - DeleteDomain
  - DescribeDomain
  - GetCallerIdentity

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Commencer à utiliser les pare-feux pour applications Web

L'exemple de code suivant illustre comment :

- Création d'une liste ACL web
- Ajouter une règle de correspondance de chaînes
- Ajouter des règles gérées
- Configurer la journalisation
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# AWS WAF Getting Started Script
# This script creates a Web ACL with a string match rule and AWS Managed Rules,
# associates it with a CloudFront distribution, and then cleans up all resources.

set -euo pipefail

# Security: Restrict file permissions
umask 077

# Set up logging with secure file handling
LOG_FILE="waf-tutorial.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1
```

```
echo "================================================="
echo "AWS WAF Getting Started Tutorial"
echo "================================================="
echo "This script will create AWS WAF resources and associate"
echo "them with a CloudFront distribution."
echo ""

# Maximum number of retries for operations
MAX_RETRIES=3

# Function to handle errors securely
handle_error() {
    local error_msg="$1"
    echo "ERROR: $error_msg" >&2
    echo "Check the log file for details: $LOG_FILE" >&2
    cleanup_resources
    exit 1
}

# Function to validate AWS CLI JSON output
validate_json() {
    local json_string="$1"
    if ! echo "$json_string" | jq empty 2>/dev/null; then
        return 1
    fi
    return 0
}

# Function to safely extract JSON values
extract_json_value() {
    local json_string="$1"
    local key_path="$2"

    if ! validate_json "$json_string"; then
        return 1
    fi

    echo "$json_string" | jq -r "$key_path" 2>/dev/null || return 1
}

# Function to clean up resources securely
cleanup_resources() {
    echo ""
```

```
    echo "================================================="
    echo "CLEANING UP RESOURCES"
    echo "================================================="

    if [ -n "${DISTRIBUTION_ID:-}" ] && [ -n "${WEB_ACL_ARN:-}" ]; then
        echo "Disassociating Web ACL from CloudFront distribution..."
        local account_id
        account_id=$(aws sts get-caller-identity --query Account --output text
 2>/dev/null) || account_id=""

        if [ -z "$account_id" ]; then
            echo "Warning: Could not retrieve AWS account ID"
            return
        fi

        local disassociate_result
        disassociate_result=$(aws wafv2 disassociate-web-acl \
            --resource-arn "arn:aws:cloudfront::${account_id}:distribution/
${DISTRIBUTION_ID}" \
            --region us-east-1 2>&1) || true

        if echo "$disassociate_result" | grep -qi "error"; then
            echo "Warning: Failed to disassociate Web ACL: $disassociate_result"
        else
            echo "Web ACL disassociated successfully."
        fi
    fi

    if [ -n "${WEB_ACL_ID:-}" ] && [ -n "${WEB_ACL_NAME:-}" ]; then
        echo "Deleting Web ACL..."

        local get_result
        get_result=$(aws wafv2 get-web-acl \
            --name "$WEB_ACL_NAME" \
            --scope CLOUDFRONT \
            --id "$WEB_ACL_ID" \
            --region us-east-1 2>&1) || true

        if echo "$get_result" | grep -qi "error"; then
            echo "Warning: Failed to get Web ACL for deletion: $get_result"
            echo "You may need to manually delete the Web ACL using the AWS
 Console."
        else
            local latest_token
```

```
            latest_token=$(extract_json_value "$get_result" '.WebACL.LockToken'
 2>/dev/null) || latest_token=""

            if [ -n "$latest_token" ]; then
                local delete_result
                delete_result=$(aws wafv2 delete-web-acl \
                    --name "$WEB_ACL_NAME" \
                    --scope CLOUDFRONT \
                    --id "$WEB_ACL_ID" \
                    --lock-token "$latest_token" \
                    --region us-east-1 2>&1) || true

                if echo "$delete_result" | grep -qi "error"; then
                    echo "Warning: Failed to delete Web ACL: $delete_result"
                    echo "You may need to manually delete the Web ACL using the
 AWS Console."
                else
                    echo "Web ACL deleted successfully."
                fi
            else
                echo "Warning: Could not extract lock token for deletion. You may
 need to manually delete the Web ACL."
            fi
        fi
    fi

    echo "Cleanup process completed."
}

# Security: Trap EXIT to ensure cleanup on any exit
trap cleanup_resources EXIT

# Generate a random identifier for resource names using secure method
RANDOM_ID=$(openssl rand -hex 4) || handle_error "Failed to generate random ID"
WEB_ACL_NAME="MyWebACL-${RANDOM_ID}"
METRIC_NAME="MyWebACLMetrics-${RANDOM_ID}"

echo "Using Web ACL name: $WEB_ACL_NAME"

# Step 1: Create a Web ACL
echo ""
echo "================================================="
echo "STEP 1: Creating Web ACL"
echo "================================================="
```

```
local create_result
create_result=$(aws wafv2 create-web-acl \
    --name "$WEB_ACL_NAME" \
    --scope "CLOUDFRONT" \
    --default-action Allow={} \
    --visibility-config
 "SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=
$METRIC_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-waf-gs \
    --region us-east-1 2>&1) || handle_error "Failed to create Web ACL"

if ! validate_json "$create_result"; then
    handle_error "Invalid JSON response from create-web-acl"
fi

# Extract Web ACL ID, ARN, and Lock Token from the response
WEB_ACL_ID=$(extract_json_value "$create_result" '.Summary.Id') || handle_error
 "Failed to extract Web ACL ID"
WEB_ACL_ARN=$(extract_json_value "$create_result" '.Summary.ARN') || handle_error
 "Failed to extract Web ACL ARN"
LOCK_TOKEN=$(extract_json_value "$create_result" '.Summary.LockToken') ||
 handle_error "Failed to extract Lock Token"

echo "Web ACL created successfully with ID: $WEB_ACL_ID"
echo "Lock Token: [REDACTED]"

# Step 2: Add a String Match Rule
echo ""
echo "================================================="
echo "STEP 2: Adding String Match Rule"
echo "================================================="

for ((i=1; i<=MAX_RETRIES; i++)); do
    echo "Attempt $i to add string match rule..."

    # Get the latest lock token before updating
    local get_result
    get_result=$(aws wafv2 get-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope CLOUDFRONT \
        --id "$WEB_ACL_ID" \
        --region us-east-1 2>&1) || true
```

```
    if echo "$get_result" | grep -qi "error"; then
        echo "Warning: Failed to get Web ACL for update: $get_result"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to get Web ACL after $MAX_RETRIES attempts"
        fi
        sleep 2
        continue
    fi

    if ! validate_json "$get_result"; then
        echo "Warning: Invalid JSON response from get-web-acl"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Invalid JSON response after $MAX_RETRIES attempts"
        fi
        sleep 2
        continue
    fi

    local latest_token
    latest_token=$(extract_json_value "$get_result" '.WebACL.LockToken' 2>/dev/
null) || true

    if [ -z "$latest_token" ]; then
        echo "Warning: Could not extract lock token for update"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to extract lock token after $MAX_RETRIES
 attempts"
        fi
        sleep 2
        continue
    fi

    local update_result
    update_result=$(aws wafv2 update-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope "CLOUDFRONT" \
        --id "$WEB_ACL_ID" \
        --lock-token "$latest_token" \
        --default-action Allow={} \
        --rules '[{
            "Name": "UserAgentRule",
            "Priority": 0,
            "Statement": {
                "ByteMatchStatement": {
```

```
                    "SearchString": "MyAgent",
                    "FieldToMatch": {
                        "SingleHeader": {
                            "Name": "user-agent"
                        }
                    },
                    "TextTransformations": [
                        {
                            "Priority": 0,
                            "Type": "NONE"
                        }
                    ],
                    "PositionalConstraint": "EXACTLY"
                }
            },
            "Action": {
                "Count": {}
            },
            "VisibilityConfig": {
                "SampledRequestsEnabled": true,
                "CloudWatchMetricsEnabled": true,
                "MetricName": "UserAgentRuleMetric"
            }
        }]' \
        --visibility-config
 "SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=
$METRIC_NAME" \
        --region us-east-1 2>&1) || true

    if echo "$update_result" | grep -qi "WAFOptimisticLockException"; then
        echo "Optimistic lock exception encountered. Will retry with new lock
 token."
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to add string match rule after $MAX_RETRIES
 attempts"
        fi
        sleep 2
        continue
    elif echo "$update_result" | grep -qi "error"; then
        handle_error "Failed to add string match rule: $update_result"
    else
        echo "String match rule added successfully."
        break
    fi
```

```
done

# Step 3: Add AWS Managed Rules
echo ""
echo "================================================="
echo "STEP 3: Adding AWS Managed Rules"
echo "================================================="

for ((i=1; i<=MAX_RETRIES; i++)); do
    echo "Attempt $i to add AWS Managed Rules..."

    # Get the latest lock token before updating
    local get_result
    get_result=$(aws wafv2 get-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope CLOUDFRONT \
        --id "$WEB_ACL_ID" \
        --region us-east-1 2>&1) || true

    if echo "$get_result" | grep -qi "error"; then
        echo "Warning: Failed to get Web ACL for update: $get_result"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to get Web ACL after $MAX_RETRIES attempts"
        fi
        sleep 2
        continue
    fi

    if ! validate_json "$get_result"; then
        echo "Warning: Invalid JSON response from get-web-acl"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Invalid JSON response after $MAX_RETRIES attempts"
        fi
        sleep 2
        continue
    fi

    local latest_token
    latest_token=$(extract_json_value "$get_result" '.WebACL.LockToken' 2>/dev/
null) || true

    if [ -z "$latest_token" ]; then
        echo "Warning: Could not extract lock token for update"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
```

```
            handle_error "Failed to extract lock token after $MAX_RETRIES
attempts"
        fi
        sleep 2
        continue
    fi

    local update_result
    update_result=$(aws wafv2 update-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope "CLOUDFRONT" \
        --id "$WEB_ACL_ID" \
        --lock-token "$latest_token" \
        --default-action Allow={} \
        --rules '[{
            "Name": "UserAgentRule",
            "Priority": 0,
            "Statement": {
                "ByteMatchStatement": {
                    "SearchString": "MyAgent",
                    "FieldToMatch": {
                        "SingleHeader": {
                            "Name": "user-agent"
                        }
                    },
                    "TextTransformations": [
                        {
                            "Priority": 0,
                            "Type": "NONE"
                        }
                    ],
                    "PositionalConstraint": "EXACTLY"
                }
            },
            "Action": {
                "Count": {}
            },
            "VisibilityConfig": {
                "SampledRequestsEnabled": true,
                "CloudWatchMetricsEnabled": true,
                "MetricName": "UserAgentRuleMetric"
            }
        },
        {
```

```
            "Name": "AWS-AWSManagedRulesCommonRuleSet",
            "Priority": 1,
            "Statement": {
                "ManagedRuleGroupStatement": {
                    "VendorName": "AWS",
                    "Name": "AWSManagedRulesCommonRuleSet",
                    "ExcludedRules": []
                }
            },
            "OverrideAction": {
                "Count": {}
            },
            "VisibilityConfig": {
                "SampledRequestsEnabled": true,
                "CloudWatchMetricsEnabled": true,
                "MetricName": "AWS-AWSManagedRulesCommonRuleSet"
            }
        }]' \
        --visibility-config
 "SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=
$METRIC_NAME" \
        --region us-east-1 2>&1) || true

    if echo "$update_result" | grep -qi "WAFOptimisticLockException"; then
        echo "Optimistic lock exception encountered. Will retry with new lock
 token."
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to add AWS Managed Rules after $MAX_RETRIES
 attempts"
        fi
        sleep 2
        continue
    elif echo "$update_result" | grep -qi "error"; then
        handle_error "Failed to add AWS Managed Rules: $update_result"
    else
        echo "AWS Managed Rules added successfully."
        break
    fi
done

# Step 4: List CloudFront distributions
echo ""
echo "================================================="
echo "STEP 4: Listing CloudFront Distributions"
```

```
echo "=================================================="

local cf_result
cf_result=$(aws cloudfront list-distributions --query "DistributionList.Items[*].
{Id:Id,DomainName:DomainName}" --output table 2>&1) || cf_result=""

if echo "$cf_result" | grep -qi "error"; then
    echo "Warning: Failed to list CloudFront distributions: $cf_result"
    echo "Continuing without CloudFront association."
    DISTRIBUTION_ID=""
else
    echo "$cf_result"

    # Auto-select first CloudFront distribution if available
    echo ""
    echo "=================================================="
    echo "STEP 5: Associate Web ACL with CloudFront Distribution"
    echo "=================================================="

    local first_dist
    first_dist=$(aws cloudfront list-distributions --query
 "DistributionList.Items[0].Id" --output text 2>&1) || first_dist=""

    if [ -n "$first_dist" ] && [ "$first_dist" != "None" ] && ! echo
 "$first_dist" | grep -qi "error"; then
        DISTRIBUTION_ID="$first_dist"
        echo "Using CloudFront distribution: $DISTRIBUTION_ID"

        local account_id
        account_id=$(aws sts get-caller-identity --query Account --output text
 2>/dev/null) || account_id=""

        if [ -z "$account_id" ]; then
            echo "Warning: Could not retrieve AWS account ID for association"
            DISTRIBUTION_ID=""
        else
            local associate_result
            associate_result=$(aws wafv2 associate-web-acl \
                --web-acl-arn "$WEB_ACL_ARN" \
                --resource-arn "arn:aws:cloudfront::${account_id}:distribution/
${DISTRIBUTION_ID}" \
                --region us-east-1 2>&1) || true

            if echo "$associate_result" | grep -qi "error"; then
```

```
                echo "Warning: Failed to associate Web ACL with CloudFront
 distribution: $associate_result"
                echo "Continuing without CloudFront association."
                DISTRIBUTION_ID=""
            else
                echo "Web ACL associated with CloudFront distribution
 successfully."
            fi
        fi
    else
        echo "No CloudFront distributions available. Skipping association."
        DISTRIBUTION_ID=""
    fi
fi

# Display summary of created resources
echo ""
echo "================================================="
echo "RESOURCE SUMMARY"
echo "================================================="
echo "Web ACL Name: $WEB_ACL_NAME"
echo "Web ACL ID: $WEB_ACL_ID"
echo "Web ACL ARN: $WEB_ACL_ARN"
if [ -n "${DISTRIBUTION_ID:-}" ]; then
    echo "Associated CloudFront Distribution: $DISTRIBUTION_ID"
fi
echo ""

# Auto-confirm cleanup
echo "================================================="
echo "CLEANUP CONFIRMATION"
echo "================================================="
echo "Proceeding with automatic cleanup of all created resources..."

echo ""
echo "================================================="
echo "Tutorial completed!"
echo "================================================="
echo "Log file: $LOG_FILE"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AssociateWebAcl

- CreateWebAcl
- DeleteWebAcl
- DisassociateWebAcl
- GetCallerIdentity
- GetWebAcl
- ListDistributions
- UpdateWebAcl

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Transférer les secrets codés en dur vers un stockage secret sécurisé

L'exemple de code suivant illustre comment :

- Création de rôles IAM
- Création d'un secret dans Secrets Manager
- Mettez à jour le code de votre application
- Mettre à jour le secret
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# Script to move hardcoded secrets to AWS Secrets Manager
```

```
# This script demonstrates how to create IAM roles, store a secret in AWS Secrets
 Manager,
# and set up appropriate permissions

set -euo pipefail

# Set up logging
LOG_FILE="secrets_manager_tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS Secrets Manager tutorial script at $(date)"
echo "============================================================"

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -qi "error"; then
        echo "ERROR: Command failed: $cmd"
        echo "$output"
        cleanup_resources
        exit 1
    fi
}

# Function to generate a random identifier using secure method
generate_random_id() {
    python3 -c "import secrets; print('sm' + secrets.token_hex(4))"
}

# Function to safely clean up resources
cleanup_resources() {
    echo ""
    echo "========================================="
    echo "RESOURCES CREATED"
    echo "========================================="

    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Secret: $SECRET_NAME"
    fi

    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "IAM Role: $RUNTIME_ROLE_NAME"
```

```
    fi

    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "IAM Role: $ADMIN_ROLE_NAME"
    fi

    echo ""
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "Cleaning up all created resources..."

    # Delete secret if it exists
    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Deleting secret: $SECRET_NAME"
        aws secretsmanager delete-secret --secret-id "$SECRET_NAME" --force-
delete-without-recovery 2>/dev/null || true
    fi

    # Detach policies and delete runtime role if it exists
    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "Deleting inline policies from runtime role: $RUNTIME_ROLE_NAME"
        for policy in $(aws iam list-role-policies --role-name
 "$RUNTIME_ROLE_NAME" --query 'PolicyNames[]' --output text 2>/dev/null || true);
 do
            aws iam delete-role-policy --role-name "$RUNTIME_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
        done
        echo "Deleting IAM role: $RUNTIME_ROLE_NAME"
        aws iam delete-role --role-name "$RUNTIME_ROLE_NAME" 2>/dev/null || true
    fi

    # Detach policies and delete admin role if it exists
    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "Detaching policy from role: $ADMIN_ROLE_NAME"
        aws iam detach-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-arn
 "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>/dev/null || true

        for policy in $(aws iam list-role-policies --role-name "$ADMIN_ROLE_NAME"
 --query 'PolicyNames[]' --output text 2>/dev/null || true); do
            aws iam delete-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
        done
```

```
        echo "Deleting IAM role: $ADMIN_ROLE_NAME"
        aws iam delete-role --role-name "$ADMIN_ROLE_NAME" 2>/dev/null || true
    fi

    echo "Cleanup completed."
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Running cleanup..."; cleanup_resources' INT TERM
 EXIT

# Generate random identifiers for resources
ADMIN_ROLE_NAME="SecretsManagerAdmin-$(generate_random_id)"
RUNTIME_ROLE_NAME="RoleToRetrieveSecretAtRuntime-$(generate_random_id)"
SECRET_NAME="MyAPIKey-$(generate_random_id)"

echo "Using the following resource names:"
echo "Admin Role: $ADMIN_ROLE_NAME"
echo "Runtime Role: $RUNTIME_ROLE_NAME"
echo "Secret Name: $SECRET_NAME"
echo ""

# Step 1: Create IAM roles
echo "Creating IAM roles..."

# Create assume role policy document
ASSUME_ROLE_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF
)

# Create the SecretsManagerAdmin role
echo "Creating admin role: $ADMIN_ROLE_NAME"
```

```
ADMIN_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$ADMIN_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$ADMIN_ROLE_OUTPUT" "create-role for admin"
echo "$ADMIN_ROLE_OUTPUT"
aws iam tag-role --role-name "$ADMIN_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs

# Attach the SecretsManagerReadWrite policy to the admin role
echo "Attaching SecretsManagerReadWrite policy to admin role"
ATTACH_POLICY_OUTPUT=$(aws iam attach-role-policy \
    --role-name "$ADMIN_ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>&1)

check_error "$ATTACH_POLICY_OUTPUT" "attach-role-policy for admin"
echo "Policy attached successfully"

# Create the RoleToRetrieveSecretAtRuntime role
echo "Creating runtime role: $RUNTIME_ROLE_NAME"
RUNTIME_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$RUNTIME_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$RUNTIME_ROLE_OUTPUT" "create-role for runtime"
echo "$RUNTIME_ROLE_OUTPUT"
aws iam tag-role --role-name "$RUNTIME_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs

# Wait for roles to be fully created
echo "Waiting for IAM roles to be fully created..."
sleep 10

# Step 2: Create a secret in AWS Secrets Manager
echo "Creating secret in AWS Secrets Manager..."

# Generate secure secret value using environment variable or secure method
# WARNING: In production, use secure methods to inject secrets (AWS CodeBuild,
 parameter store, etc.)
if [ -z "${TUTORIAL_SECRET_VALUE:-}" ]; then
    SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_client_id','ClientSecret':__import__('secrets').token_u
else
    SECRET_VALUE="$TUTORIAL_SECRET_VALUE"
```

```
fi

CREATE_SECRET_OUTPUT=$(aws secretsmanager create-secret \
    --name "$SECRET_NAME" \
    --description "API key for my application" \
    --secret-string "$SECRET_VALUE" \
    --add-replica-regions 'Region=us-east-1' \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-secrets-manager-gs
 2>&1)

check_error "$CREATE_SECRET_OUTPUT" "create-secret"
echo "$CREATE_SECRET_OUTPUT"

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text 2>&1)
check_error "$ACCOUNT_ID" "get-caller-identity"
echo "Account ID: $ACCOUNT_ID"

# Get secret ARN for precise resource policy
echo "Getting secret ARN..."
SECRET_ARN=$(aws secretsmanager describe-secret \
    --secret-id "$SECRET_NAME" \
    --query 'ARN' \
    --output text 2>&1)
check_error "$SECRET_ARN" "describe-secret"

# Add resource policy to the secret with least privilege
echo "Adding resource policy to secret..."
RESOURCE_POLICY=$(cat <<EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRuntimeRoleReadSecret",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::$ACCOUNT_ID:role/$RUNTIME_ROLE_NAME"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "$SECRET_ARN",
            "Condition": {
                "StringEquals": {
                    "secretsmanager:VersionStage": "AWSCURRENT"
```

```
                }
            }
        }
    ]
}
EOF
)

PUT_POLICY_OUTPUT=$(aws secretsmanager put-resource-policy \
    --secret-id "$SECRET_NAME" \
    --resource-policy "$RESOURCE_POLICY" \
    --block-public-policy 2>&1)

check_error "$PUT_POLICY_OUTPUT" "put-resource-policy"
echo "Resource policy added successfully"

# Enable rotation policy recommendation
echo "Enabling secret metadata tags for rotation tracking..."
aws secretsmanager tag-resource \
    --secret-id "$SECRET_NAME" \
    --tags Key=Purpose,Value=Tutorial Key=AutoRotation,Value=Recommended 2>/dev/
null || true

# Step 3: Demonstrate retrieving the secret
echo "Retrieving the secret value (for demonstration purposes)..."
GET_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$GET_SECRET_OUTPUT" "get-secret-value"
echo "Secret retrieved successfully. Secret metadata:"
echo "$GET_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"

# Step 4: Update the secret with new values
echo "Updating the secret with new values..."
UPDATE_SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_new_client_id','ClientSecret':__import__('secrets').tok

UPDATE_SECRET_OUTPUT=$(aws secretsmanager update-secret \
    --secret-id "$SECRET_NAME" \
    --secret-string "$UPDATE_SECRET_VALUE" 2>&1)

check_error "$UPDATE_SECRET_OUTPUT" "update-secret"
```

```
echo "Secret updated successfully"

# Step 5: Verify the updated secret
echo "Verifying the updated secret..."
VERIFY_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$VERIFY_SECRET_OUTPUT" "get-secret-value for verification"
echo "Updated secret retrieved successfully. Secret metadata:"
echo "$VERIFY_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"

# Step 6: Display rotation recommendations
echo ""
echo "Rotation Configuration Recommendations:"
echo "======================================="
DESCRIBE_OUTPUT=$(aws secretsmanager describe-secret --secret-id "$SECRET_NAME"
 2>&1)
if echo "$DESCRIBE_OUTPUT" | grep -q "RotationRules"; then
    echo "Current rotation configuration:"
    echo "$DESCRIBE_OUTPUT" | jq '.RotationRules' 2>/dev/null || echo "Rotation
 rules available"
else
    echo "No automatic rotation configured. Consider enabling rotation with:"
    echo "aws secretsmanager rotate-secret --secret-id $SECRET_NAME --rotation-
lambda-arn arn:aws:lambda:REGION:ACCOUNT:function:FUNCTION_NAME --rotation-rules
 AutomaticallyAfterDays=30"
fi

echo ""
echo "=================================================="
echo "Tutorial completed successfully!"
echo ""
echo "Summary of what we did:"
echo "1. Created IAM roles for managing and retrieving secrets"
echo "2. Created a secret in AWS Secrets Manager with secure generation"
echo "3. Added a least-privilege resource policy to control access to the secret"
echo "4. Retrieved the secret value (simulating application access)"
echo "5. Updated the secret with cryptographically secure values"
echo "6. Verified the updated secret"
echo ""
echo "Security best practices applied:"
echo "- Used cryptographically secure random ID generation"
```

```
echo "- Applied least-privilege resource policies with version stages"
echo "- Tagged resources for rotation tracking"
echo "- Blocked public access to secrets"
echo "- Used ARN-specific permissions instead of wildcards"
echo ""
echo "Next steps you might want to consider:"
echo "- Enable automatic secret rotation with AWS Lambda"
echo "- Implement secret caching in your application"
echo "- Set up CloudTrail logging for secret access auditing"
echo "- Use AWS CodeGuru Reviewer to find hardcoded secrets in your code"
echo "- For multi-region applications, replicate your secrets across regions"
echo "- Configure VPC endpoints for private access to Secrets Manager"
echo ""

echo "Script completed at $(date)"
exit 0
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreateRole
  - CreateSecret
  - DeleteRole
  - DeleteSecret
  - DetachRolePolicy
  - GetCallerIdentity
  - GetSecretValue
  - PutResourcePolicy
  - UpdateSecret

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Exécutez des tests de stress du processeur sur des instances de machines virtuelles à l'aide de l'injection de défauts

L'exemple de code suivant illustre comment :

- Création de rôles IAM
- Création d'une CloudWatch alarme
- Création d'un modèle d'expérience
- Exécutez l'expérience
- Vérifiez les résultats
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# AWS FIS CPU Stress Test Tutorial Script
# This script automates the steps in the AWS FIS CPU stress test tutorial

#    approach using epoch time calculations that work across all Linux
 distributions

# Set up logging
LOG_FILE="fis-tutorial-$(date +%Y%m%d-%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS FIS CPU Stress Test Tutorial Script"
echo "Logging to $LOG_FILE"
echo "============================================="

# Function to check for errors in command output
```

```
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -i "error" > /dev/null; then
        # Ignore specific expected errors
        if [[ "$cmd" == *"aws fis get-experiment"* ]] && [[ "$output" ==
 *"ConfigurationFailure"* ]]; then
            echo "Note: Experiment failed due to configuration issue. This is
 expected in some cases."
            return 0
        fi

        echo "ERROR: Command failed: $cmd"
        echo "Output: $output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Cleaning up resources..."

    if [ -n "$EXPERIMENT_ID" ]; then
        echo "Stopping experiment $EXPERIMENT_ID if running..."
        aws fis stop-experiment --id "$EXPERIMENT_ID" 2>/dev/null || true
    fi

    if [ -n "$TEMPLATE_ID" ]; then
        echo "Deleting experiment template $TEMPLATE_ID..."
        aws fis delete-experiment-template --id "$TEMPLATE_ID" || true
    fi

    if [ -n "$INSTANCE_ID" ]; then
        echo "Terminating EC2 instance $INSTANCE_ID..."
        aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || true
    fi

    if [ -n "$ALARM_NAME" ]; then
        echo "Deleting CloudWatch alarm $ALARM_NAME..."
        aws cloudwatch delete-alarms --alarm-names "$ALARM_NAME" || true
    fi
```

```
    if [ -n "$INSTANCE_PROFILE_NAME" ]; then
        echo "Removing role from instance profile..."
        aws iam remove-role-from-instance-profile --instance-profile-name
 "$INSTANCE_PROFILE_NAME" --role-name "$EC2_ROLE_NAME" || true

        echo "Deleting instance profile..."
        aws iam delete-instance-profile --instance-profile-name
 "$INSTANCE_PROFILE_NAME" || true
    fi

    if [ -n "$FIS_ROLE_NAME" ]; then
        echo "Deleting FIS role policy..."
        aws iam delete-role-policy --role-name "$FIS_ROLE_NAME" --policy-name
 "$FIS_POLICY_NAME" || true

        echo "Deleting FIS role..."
        aws iam delete-role --role-name "$FIS_ROLE_NAME" || true
    fi

    if [ -n "$EC2_ROLE_NAME" ]; then
        echo "Detaching policy from EC2 role..."
        aws iam detach-role-policy --role-name "$EC2_ROLE_NAME" --policy-arn
 "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore" || true

        echo "Deleting EC2 role..."
        aws iam delete-role --role-name "$EC2_ROLE_NAME" || true
    fi

    echo "Cleanup completed."
}

# Generate unique identifiers for resources
TIMESTAMP=$(date +%Y%m%d%H%M%S)
FIS_ROLE_NAME="FISRole-${TIMESTAMP}"
FIS_POLICY_NAME="FISPolicy-${TIMESTAMP}"
EC2_ROLE_NAME="EC2SSMRole-${TIMESTAMP}"
INSTANCE_PROFILE_NAME="EC2SSMProfile-${TIMESTAMP}"
ALARM_NAME="FIS-CPU-Alarm-${TIMESTAMP}"

# Track created resources
CREATED_RESOURCES=()

echo "Step 1: Creating IAM role for AWS FIS"
# Create trust policy file for AWS FIS
```

```
cat > fis-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "fis.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role for FIS
echo "Creating IAM role $FIS_ROLE_NAME for AWS FIS..."
FIS_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$FIS_ROLE_NAME" \
  --assume-role-policy-document file://fis-trust-policy.json)
check_error "$FIS_ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$FIS_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $FIS_ROLE_NAME")

# Create policy document for SSM actions
cat > fis-ssm-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:SendCommand",
        "ssm:ListCommands",
        "ssm:ListCommandInvocations"
      ],
      "Resource": "*"
    }
  ]
}
EOF

# Attach policy to the role
```

```
echo "Attaching policy $FIS_POLICY_NAME to role $FIS_ROLE_NAME..."
FIS_POLICY_OUTPUT=$(aws iam put-role-policy \
  --role-name "$FIS_ROLE_NAME" \
  --policy-name "$FIS_POLICY_NAME" \
  --policy-document file://fis-ssm-policy.json)
check_error "$FIS_POLICY_OUTPUT" "aws iam put-role-policy"
CREATED_RESOURCES+=("IAM Policy: $FIS_POLICY_NAME attached to $FIS_ROLE_NAME")

echo "Step 2: Creating IAM role for EC2 instance with SSM permissions"
# Create trust policy file for EC2
cat > ec2-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role for EC2
echo "Creating IAM role $EC2_ROLE_NAME for EC2 instance..."
EC2_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$EC2_ROLE_NAME" \
  --assume-role-policy-document file://ec2-trust-policy.json)
check_error "$EC2_ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$EC2_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $EC2_ROLE_NAME")

# Attach SSM policy to the EC2 role
echo "Attaching AmazonSSMManagedInstanceCore policy to role $EC2_ROLE_NAME..."
EC2_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$EC2_ROLE_NAME" \
  --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore)
check_error "$EC2_POLICY_OUTPUT" "aws iam attach-role-policy"
CREATED_RESOURCES+=("IAM Policy: AmazonSSMManagedInstanceCore attached to
 $EC2_ROLE_NAME")
```

```
# Create instance profile
echo "Creating instance profile $INSTANCE_PROFILE_NAME..."
PROFILE_OUTPUT=$(aws iam create-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME")
check_error "$PROFILE_OUTPUT" "aws iam create-instance-profile"
CREATED_RESOURCES+=("IAM Instance Profile: $INSTANCE_PROFILE_NAME")

# Add role to instance profile
echo "Adding role $EC2_ROLE_NAME to instance profile $INSTANCE_PROFILE_NAME..."
ADD_ROLE_OUTPUT=$(aws iam add-role-to-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME" \
  --role-name "$EC2_ROLE_NAME")
check_error "$ADD_ROLE_OUTPUT" "aws iam add-role-to-instance-profile"

# Wait for role to propagate
echo "Waiting for IAM role to propagate..."
sleep 10

echo "Step 3: Launching EC2 instance"
# Get the latest Amazon Linux 2 AMI ID
echo "Finding latest Amazon Linux 2 AMI..."
AMI_ID=$(aws ec2 describe-images \
  --owners amazon \
  --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
  --query "sort_by(Images, &CreationDate)[-1].ImageId" \
  --output text)
check_error "$AMI_ID" "aws ec2 describe-images"
echo "Using AMI: $AMI_ID"

# Launch EC2 instance
echo "Launching EC2 instance with AMI $AMI_ID..."
INSTANCE_OUTPUT=$(aws ec2 run-instances \
  --image-id "$AMI_ID" \
  --instance-type t2.micro \
  --iam-instance-profile Name="$INSTANCE_PROFILE_NAME" \
  --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=FIS-Test-
Instance},{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-fault-injection-
service-gs}]')
check_error "$INSTANCE_OUTPUT" "aws ec2 run-instances"

# Get instance ID
INSTANCE_ID=$(echo "$INSTANCE_OUTPUT" | grep -i "InstanceId" | head -1 | awk -
F'"' '{print $4}')
```

```
if [ -z "$INSTANCE_ID" ]; then
    echo "Failed to get instance ID"
    cleanup_on_error
    exit 1
fi
echo "Launched instance: $INSTANCE_ID"
CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")

# Enable detailed monitoring
echo "Enabling detailed monitoring for instance $INSTANCE_ID..."
MONITOR_OUTPUT=$(aws ec2 monitor-instances --instance-ids "$INSTANCE_ID")
check_error "$MONITOR_OUTPUT" "aws ec2 monitor-instances"

# Wait for instance to be running and status checks to pass
echo "Waiting for instance to be ready..."
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"
aws ec2 wait instance-status-ok --instance-ids "$INSTANCE_ID"
echo "Instance is ready"

echo "Step 4: Creating CloudWatch alarm for CPU utilization"
# Create CloudWatch alarm
echo "Creating CloudWatch alarm $ALARM_NAME..."
ALARM_OUTPUT=$(aws cloudwatch put-metric-alarm \
  --alarm-name "$ALARM_NAME" \
  --alarm-description "Alarm when CPU exceeds 50%" \
  --metric-name CPUUtilization \
  --namespace AWS/EC2 \
  --statistic Maximum \
  --period 60 \
  --threshold 50 \
  --comparison-operator GreaterThanOrEqualToThreshold \
  --dimensions "Name=InstanceId,Value=$INSTANCE_ID" \
  --evaluation-periods 1)
check_error "$ALARM_OUTPUT" "aws cloudwatch put-metric-alarm"
CREATED_RESOURCES+=("CloudWatch Alarm: $ALARM_NAME")

# Get the alarm ARN
echo "Getting CloudWatch alarm ARN..."
ALARM_ARN_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
check_error "$ALARM_ARN_OUTPUT" "aws cloudwatch describe-alarms"
ALARM_ARN=$(echo "$ALARM_ARN_OUTPUT" | grep -i "AlarmArn" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$ALARM_ARN" ]; then
```

```
    echo "Failed to get alarm ARN"
    cleanup_on_error
    exit 1
fi
echo "Alarm ARN: $ALARM_ARN"

# Wait for the alarm to initialize and reach OK state
echo "Waiting for CloudWatch alarm to initialize (60 seconds)..."
sleep 60

# Check alarm state
echo "Checking alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 | awk -
F'"' '{print $4}')
echo "Current alarm state: $ALARM_STATE"

# If alarm is not in OK state, wait longer or generate some baseline metrics
if [ "$ALARM_STATE" != "OK" ]; then
    echo "Alarm not in OK state. Waiting for alarm to stabilize (additional 60
 seconds)..."
    sleep 60

    # Check alarm state again
    ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
      --alarm-names "$ALARM_NAME")
    ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 |
 awk -F'"' '{print $4}')
    echo "Updated alarm state: $ALARM_STATE"

    if [ "$ALARM_STATE" != "OK" ]; then
        echo "Warning: Alarm still not in OK state. Experiment may fail to
 start."
    fi
fi

echo "Step 5: Creating AWS FIS experiment template"
# Get the IAM role ARN
echo "Getting IAM role ARN for $FIS_ROLE_NAME..."
ROLE_ARN_OUTPUT=$(aws iam get-role \
  --role-name "$FIS_ROLE_NAME")
check_error "$ROLE_ARN_OUTPUT" "aws iam get-role"
```

```
ROLE_ARN=$(echo "$ROLE_ARN_OUTPUT" | grep -i "Arn" | head -1 | awk -F'"' '{print
 $4}')
if [ -z "$ROLE_ARN" ]; then
    echo "Failed to get role ARN"
    cleanup_on_error
    exit 1
fi
echo "Role ARN: $ROLE_ARN"

# Get account ID and region
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"  # Default to us-east-1 if region not set
fi
INSTANCE_ARN="arn:aws:ec2:${REGION}:${ACCOUNT_ID}:instance/${INSTANCE_ID}"
echo "Instance ARN: $INSTANCE_ARN"

# Create experiment template - Fixed JSON escaping issue
cat > experiment-template.json << EOF
{
  "description": "Test CPU stress predefined SSM document",
  "targets": {
    "testInstance": {
      "resourceType": "aws:ec2:instance",
      "resourceArns": ["$INSTANCE_ARN"],
      "selectionMode": "ALL"
    }
  },
  "actions": {
    "runCpuStress": {
      "actionId": "aws:ssm:send-command",
      "parameters": {
        "documentArn": "arn:aws:ssm:$REGION::document/AWSFIS-Run-CPU-Stress",
        "documentParameters": "{\"DurationSeconds\":\"120\"}",
        "duration": "PT5M"
      },
      "targets": {
        "Instances": "testInstance"
      }
    }
  },
  "stopConditions": [
    {
```

```
      "source": "aws:cloudwatch:alarm",
      "value": "$ALARM_ARN"
    }
  ],
  "roleArn": "$ROLE_ARN",
  "tags": {
    "Name": "FIS-CPU-Stress-Experiment",
    "project": "doc-smith",
    "tutorial": "aws-fault-injection-service-gs"
  }
}
EOF

# Create experiment template
echo "Creating AWS FIS experiment template..."
TEMPLATE_OUTPUT=$(aws fis create-experiment-template --cli-input-json file://
experiment-template.json)
check_error "$TEMPLATE_OUTPUT" "aws fis create-experiment-template"
TEMPLATE_ID=$(echo "$TEMPLATE_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$TEMPLATE_ID" ]; then
    echo "Failed to get template ID"
    cleanup_on_error
    exit 1
fi
echo "Experiment template created with ID: $TEMPLATE_ID"
CREATED_RESOURCES+=("FIS Experiment Template: $TEMPLATE_ID")

echo "Step 6: Starting the experiment"
# Start the experiment
echo "Starting AWS FIS experiment using template $TEMPLATE_ID..."
EXPERIMENT_OUTPUT=$(aws fis start-experiment \
  --experiment-template-id "$TEMPLATE_ID" \
  --tags '{"Name": "FIS-CPU-Stress-Run"}')
check_error "$EXPERIMENT_OUTPUT" "aws fis start-experiment"
EXPERIMENT_ID=$(echo "$EXPERIMENT_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$EXPERIMENT_ID" ]; then
    echo "Failed to get experiment ID"
    cleanup_on_error
    exit 1
fi
echo "Experiment started with ID: $EXPERIMENT_ID"
CREATED_RESOURCES+=("FIS Experiment: $EXPERIMENT_ID")
```

```
echo "Step 7: Tracking experiment progress"
# Track experiment progress
echo "Tracking experiment progress..."
MAX_CHECKS=30
CHECK_COUNT=0
EXPERIMENT_STATE=""

while [ $CHECK_COUNT -lt $MAX_CHECKS ]; do
    EXPERIMENT_INFO=$(aws fis get-experiment --id "$EXPERIMENT_ID")
    # Don't check for errors here, as we expect some experiments to fail

    EXPERIMENT_STATE=$(echo "$EXPERIMENT_INFO" | grep -i "status" | head -1 | awk
 -F'"' '{print $4}')
    echo "Experiment state: $EXPERIMENT_STATE"

    if [ "$EXPERIMENT_STATE" == "completed" ] || [ "$EXPERIMENT_STATE" ==
 "stopped" ] || [ "$EXPERIMENT_STATE" == "failed" ]; then
        # Show the reason for the state
        REASON=$(echo "$EXPERIMENT_INFO" | grep -i "reason" | head -1 | awk -F'"'
 '{print $4}')
        if [ -n "$REASON" ]; then
            echo "Reason: $REASON"
        fi
        break
    fi

    echo "Waiting 10 seconds before checking again..."
    sleep 10
    CHECK_COUNT=$((CHECK_COUNT + 1))
done

if [ $CHECK_COUNT -eq $MAX_CHECKS ]; then
    echo "Experiment is taking longer than expected. You can check its status
 later using:"
    echo "aws fis get-experiment --id $EXPERIMENT_ID"
fi

echo "Step 8: Verifying experiment results"
# Check CloudWatch alarm state
echo "Checking CloudWatch alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms --alarm-names "$ALARM_NAME")
check_error "$ALARM_STATE_OUTPUT" "aws cloudwatch describe-alarms"
echo "$ALARM_STATE_OUTPUT"
```

```
# Get CPU utilization metrics
echo "Getting CPU utilization metrics..."
END_TIME=$(date -u +"%Y-%m-%dT%H:%M:%SZ")

# FIXED: Cross-platform compatible way to calculate time 10 minutes ago
# This approach uses epoch seconds and basic arithmetic which works on all Linux
 distributions
CURRENT_EPOCH=$(date +%s)
TEN_MINUTES_AGO_EPOCH=$((CURRENT_EPOCH - 600))
START_TIME=$(date -u -d "@$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ" 2>/dev/
null || date -u -r "$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ")

# Create metric query file
cat > metric-query.json << EOF
[
  {
    "Id": "cpu",
    "MetricStat": {
      "Metric": {
        "Namespace": "AWS/EC2",
        "MetricName": "CPUUtilization",
        "Dimensions": [
          {
            "Name": "InstanceId",
            "Value": "$INSTANCE_ID"
          }
        ]
      },
      "Period": 60,
      "Stat": "Maximum"
    }
  }
]
EOF

METRICS_OUTPUT=$(aws cloudwatch get-metric-data \
  --start-time "$START_TIME" \
  --end-time "$END_TIME" \
  --metric-data-queries file://metric-query.json)
check_error "$METRICS_OUTPUT" "aws cloudwatch get-metric-data"
echo "CPU Utilization Metrics:"
echo "$METRICS_OUTPUT"
```

```
# Display summary of created resources
echo ""
echo "============================================="
echo "RESOURCES CREATED"
echo "============================================="
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource"
done
echo "============================================="

# Prompt for cleanup
echo ""
echo "============================================="
echo "CLEANUP CONFIRMATION"
echo "============================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Starting cleanup process..."

    # Stop experiment if still running
    if [ "$EXPERIMENT_STATE" != "completed" ] && [ "$EXPERIMENT_STATE" !=
 "stopped" ] && [ "$EXPERIMENT_STATE" != "failed" ]; then
        echo "Stopping experiment $EXPERIMENT_ID..."
        STOP_OUTPUT=$(aws fis stop-experiment --id "$EXPERIMENT_ID")
        check_error "$STOP_OUTPUT" "aws fis stop-experiment"
        echo "Waiting for experiment to stop..."
        sleep 10
    fi

    # Delete experiment template
    echo "Deleting experiment template $TEMPLATE_ID..."
    DELETE_TEMPLATE_OUTPUT=$(aws fis delete-experiment-template --id
 "$TEMPLATE_ID")
    check_error "$DELETE_TEMPLATE_OUTPUT" "aws fis delete-experiment-template"

    # Delete CloudWatch alarm
    echo "Deleting CloudWatch alarm $ALARM_NAME..."
    DELETE_ALARM_OUTPUT=$(aws cloudwatch delete-alarms --alarm-names
 "$ALARM_NAME")
    check_error "$DELETE_ALARM_OUTPUT" "aws cloudwatch delete-alarms"

    # Terminate EC2 instance
```

```
    echo "Terminating EC2 instance $INSTANCE_ID..."
    TERMINATE_OUTPUT=$(aws ec2 terminate-instances --instance-ids "$INSTANCE_ID")
    check_error "$TERMINATE_OUTPUT" "aws ec2 terminate-instances"
    echo "Waiting for instance to terminate..."
    aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"

    # Clean up IAM resources
    echo "Removing role from instance profile..."
    REMOVE_ROLE_OUTPUT=$(aws iam remove-role-from-instance-profile \
      --instance-profile-name "$INSTANCE_PROFILE_NAME" \
      --role-name "$EC2_ROLE_NAME")
    check_error "$REMOVE_ROLE_OUTPUT" "aws iam remove-role-from-instance-profile"

    echo "Deleting instance profile..."
    DELETE_PROFILE_OUTPUT=$(aws iam delete-instance-profile \
      --instance-profile-name "$INSTANCE_PROFILE_NAME")
    check_error "$DELETE_PROFILE_OUTPUT" "aws iam delete-instance-profile"

    echo "Deleting FIS role policy..."
    DELETE_POLICY_OUTPUT=$(aws iam delete-role-policy \
      --role-name "$FIS_ROLE_NAME" \
      --policy-name "$FIS_POLICY_NAME")
    check_error "$DELETE_POLICY_OUTPUT" "aws iam delete-role-policy"

    echo "Detaching policy from EC2 role..."
    DETACH_POLICY_OUTPUT=$(aws iam detach-role-policy \
      --role-name "$EC2_ROLE_NAME" \
      --policy-arn "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore")
    check_error "$DETACH_POLICY_OUTPUT" "aws iam detach-role-policy"

    echo "Deleting FIS role..."
    DELETE_FIS_ROLE_OUTPUT=$(aws iam delete-role \
      --role-name "$FIS_ROLE_NAME")
    check_error "$DELETE_FIS_ROLE_OUTPUT" "aws iam delete-role"

    echo "Deleting EC2 role..."
    DELETE_EC2_ROLE_OUTPUT=$(aws iam delete-role \
      --role-name "$EC2_ROLE_NAME")
    check_error "$DELETE_EC2_ROLE_OUTPUT" "aws iam delete-role"

    # Clean up temporary files
    echo "Cleaning up temporary files..."
    rm -f fis-trust-policy.json ec2-trust-policy.json fis-ssm-policy.json
  experiment-template.json metric-query.json
```

```
    echo "Cleanup completed successfully."
else
    echo "Cleanup skipped. Resources will remain in your AWS account."
    echo "You can manually clean up the resources listed above."
fi

echo ""
echo "Script execution completed."
echo "Log file: $LOG_FILE"
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - CreateExperimentTemplate
  - CreateInstanceProfile
  - CreateRole
  - DeleteAlarms
  - DeleteExperimentTemplate
  - DeleteInstanceProfile
  - DeleteRole
  - DeleteRolePolicy
  - DescribeAlarms
  - DescribeImages
  - DetachRolePolicy
  - GetCallerIdentity
  - GetExperiment
  - GetMetricData
  - GetRole
  - MonitorInstances
  - PutMetricAlarm
  - PutRolePolicy

- RemoveRoleFromInstanceProfile
- RunInstances
- StartExperiment
- StopExperiment
- TerminateInstances
- Attente

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Configuration de la gestion des systèmes

L'exemple de code suivant illustre comment :

- Création d'autorisations IAM pour Systems Manager
- Création d'un rôle IAM pour Systems Manager
- Configuration de Systems Manager
- Vérifiez la configuration
- nettoyer des ressources ;

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code.

```
#!/bin/bash

# AWS Systems Manager Setup Script
# This script sets up AWS Systems Manager for a single account and region
#
```

```
# Version 17 fixes:
# 1. Added cloudformation.amazonaws.com to the IAM role trust policy
# 2. Systems Manager Quick Setup uses CloudFormation for deployments, so the role
 must trust CloudFormation service

# Initialize log file
LOG_FILE="ssm_setup_$(date +%Y%m%d_%H%M%S).log"
UNIQUE_ID=$(openssl rand -hex 4)
echo "Starting AWS Systems Manager setup at $(date)" > "$LOG_FILE"

# Function to log commands and their outputs with immediate terminal display
log_cmd() {
    echo "$(date): Running command: $1" | tee -a "$LOG_FILE"
    local output
    output=$(eval "$1" 2>&1)
    local status=$?
    echo "$output" | tee -a "$LOG_FILE"
    return $status
}

# Function to check for errors in command output
check_error() {
    local cmd_output="$1"
    local cmd_status="$2"
    local error_msg="$3"

    if [[ $cmd_status -ne 0 || "$cmd_output" =~ [Ee][Rr][Rr][Oo][Rr] ]]; then
        echo "ERROR: $error_msg" | tee -a "$LOG_FILE"
        echo "Command output: $cmd_output" | tee -a "$LOG_FILE"
        cleanup_on_error
        exit 1
    fi
}

# Array to track created resources for cleanup
declare -a CREATED_RESOURCES

# Function to add a resource to the tracking array
track_resource() {
    local resource_type="$1"
    local resource_id="$2"
    CREATED_RESOURCES+=("$resource_type:$resource_id")
    echo "Tracked resource: $resource_type:$resource_id" | tee -a "$LOG_FILE"
}
```

```
# Function to clean up resources on error
cleanup_on_error() {
    echo "" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"
    echo "ERROR OCCURRED - CLEANING UP RESOURCES" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"
    echo "The following resources were created:" | tee -a "$LOG_FILE"

    # Display resources in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        echo "${CREATED_RESOURCES[$i]}" | tee -a "$LOG_FILE"
    done

    echo "" | tee -a "$LOG_FILE"
    echo "Attempting to clean up resources..." | tee -a "$LOG_FILE"

    # Clean up resources in reverse order
    cleanup_resources
}

# Function to clean up all created resources
cleanup_resources() {
    # Process resources in reverse order (last created, first deleted)
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        IFS=':' read -r resource_type resource_id <<< "${CREATED_RESOURCES[$i]}"

        echo "Deleting $resource_type: $resource_id" | tee -a "$LOG_FILE"

        case "$resource_type" in
            "IAM_POLICY")
                # Delete the policy (detachment should have been handled when the
 role was deleted)
                log_cmd "aws iam delete-policy --policy-arn $resource_id" || true
                ;;
            "IAM_ROLE")
                # Detach all policies from the role first
                if [[ -n "$POLICY_ARN" ]]; then
                    log_cmd "aws iam detach-role-policy --role-name $resource_id
 --policy-arn $POLICY_ARN" || true
                fi

                # Delete the role
                log_cmd "aws iam delete-role --role-name $resource_id" || true
```

```
                ;;
            "SSM_CONFIG_MANAGER")
                log_cmd "aws ssm-quicksetup delete-configuration-manager --
manager-arn $resource_id" || true
                ;;
            *)
                echo "Unknown resource type: $resource_type, cannot delete
 automatically" | tee -a "$LOG_FILE"
                ;;
        esac
    done

    echo "Cleanup completed" | tee -a "$LOG_FILE"

    # Clean up temporary files
    rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/
null || true
}

# Main script execution
echo "AWS Systems Manager Setup Script"
echo "================================="
echo "This script will set up AWS Systems Manager for a single account and
 region."
echo "It will create IAM policies and roles, then enable Systems Manager
 features."
echo ""

# Get the current AWS region
CURRENT_REGION=$(aws configure get region)
if [[ -z "$CURRENT_REGION" ]]; then
    echo "No AWS region configured. Please specify a region:"
    CURRENT_REGION="${AWS_DEFAULT_REGION:-us-west-2}"
    if [[ -z "$CURRENT_REGION" ]]; then
        echo "ERROR: A region must be specified" | tee -a "$LOG_FILE"
        exit 1
    fi
fi

echo "Using AWS region: $CURRENT_REGION" | tee -a "$LOG_FILE"

# Step 1: Create IAM policy for Systems Manager onboarding
echo "Step 1: Creating IAM policy for Systems Manager onboarding..."
```

```
# Create policy document
cat > ssm-onboarding-policy.json << 'EOF'
{
   "Version":"2012-10-17",
   "Statement": [
     {
       "Sid": "QuickSetupActions",
       "Effect": "Allow",
       "Action": [
         "ssm-quicksetup:*"
       ],
       "Resource": "*"
     },
     {
       "Sid": "SsmReadOnly",
       "Effect": "Allow",
       "Action": [
         "ssm:DescribeAutomationExecutions",
         "ssm:GetAutomationExecution",
         "ssm:ListAssociations",
         "ssm:DescribeAssociation",
         "ssm:ListDocuments",
         "ssm:ListResourceDataSync",
         "ssm:DescribePatchBaselines",
         "ssm:GetPatchBaseline",
         "ssm:DescribeMaintenanceWindows",
         "ssm:DescribeMaintenanceWindowTasks"
       ],
       "Resource": "*"
     },
     {
       "Sid": "SsmDocument",
       "Effect": "Allow",
       "Action": [
         "ssm:GetDocument",
         "ssm:DescribeDocument"
       ],
       "Resource": [
         "arn:aws:ssm:*:*:document/AWSQuickSetupType-*",
         "arn:aws:ssm:*:*:document/AWS-EnableExplorer"
       ]
     },
     {
       "Sid": "SsmEnableExplorer",
```

```
      "Effect": "Allow",
      "Action": "ssm:StartAutomationExecution",
      "Resource": "arn:aws:ssm:*:*:automation-definition/AWS-EnableExplorer:*"
    },
    {
      "Sid": "SsmExplorerRds",
      "Effect": "Allow",
      "Action": [
        "ssm:GetOpsSummary",
        "ssm:CreateResourceDataSync",
        "ssm:UpdateResourceDataSync"
      ],
      "Resource": "arn:aws:ssm:*:*:resource-data-sync/AWS-QuickSetup-*"
    },
    {
      "Sid": "OrgsReadOnly",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListRoots",
        "organizations:ListParents",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAWSServiceAccessForOrganization"
      ],
      "Resource": "*"
    },
    {
      "Sid": "OrgsAdministration",
      "Effect": "Allow",
      "Action": [
        "organizations:EnableAWSServiceAccess",
        "organizations:RegisterDelegatedAdministrator",
        "organizations:DeregisterDelegatedAdministrator"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "organizations:ServicePrincipal": [
            "ssm.amazonaws.com",
            "ssm-quicksetup.amazonaws.com",
            "member.org.stacksets.cloudformation.amazonaws.com",
```

```
            "resource-explorer-2.amazonaws.com"
          ]
        }
      }
    },
    {
      "Sid": "CfnReadOnly",
      "Effect": "Allow",
      "Action": [
        "cloudformation:ListStacks",
        "cloudformation:DescribeStacks",
        "cloudformation:ListStackSets",
        "cloudformation:DescribeOrganizationsAccess"
      ],
      "Resource": "*"
    },
    {
      "Sid": "OrgCfnAccess",
      "Effect": "Allow",
      "Action": [
        "cloudformation:ActivateOrganizationsAccess"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CfnStackActions",
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStack",
        "cloudformation:DeleteStack",
        "cloudformation:DescribeStackResources",
        "cloudformation:DescribeStackEvents",
        "cloudformation:GetTemplate",
        "cloudformation:RollbackStack",
        "cloudformation:TagResource",
        "cloudformation:UntagResource",
        "cloudformation:UpdateStack"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stack/StackSet-AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:stack/AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:type/resource/*"
      ]
    },
```

```
    {
      "Sid": "CfnStackSetActions",
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStackInstances",
        "cloudformation:CreateStackSet",
        "cloudformation:DeleteStackInstances",
        "cloudformation:DeleteStackSet",
        "cloudformation:DescribeStackInstance",
        "cloudformation:DetectStackSetDrift",
        "cloudformation:ListStackInstanceResourceDrifts",
        "cloudformation:DescribeStackSet",
        "cloudformation:DescribeStackSetOperation",
        "cloudformation:ListStackInstances",
        "cloudformation:ListStackSetOperations",
        "cloudformation:ListStackSetOperationResults",
        "cloudformation:TagResource",
        "cloudformation:UntagResource",
        "cloudformation:UpdateStackSet"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:type/resource/*",
        "arn:aws:cloudformation:*:*:stackset-target/AWS-QuickSetup-*:*"
      ]
    },
    {
      "Sid": "ValidationReadonlyActions",
      "Effect": "Allow",
      "Action": [
        "iam:ListRoles",
        "iam:GetRole"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IamRolesMgmt",
      "Effect": "Allow",
      "Action": [
        "iam:CreateRole",
        "iam:DeleteRole",
        "iam:GetRole",
        "iam:AttachRolePolicy",
        "iam:DetachRolePolicy",
```

```
          "iam:GetRolePolicy",
          "iam:ListRolePolicies"
        ],
        "Resource": [
          "arn:aws:iam::*:role/AWS-QuickSetup-*",
          "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
        ]
      },
      {
        "Sid": "IamPassRole",
        "Effect": "Allow",
        "Action": [
          "iam:PassRole"
        ],
        "Resource": [
          "arn:aws:iam::*:role/AWS-QuickSetup-*",
          "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
        ],
        "Condition": {
          "StringEquals": {
            "iam:PassedToService": [
              "ssm.amazonaws.com",
              "ssm-quicksetup.amazonaws.com",
              "cloudformation.amazonaws.com"
            ]
          }
        }
      },
      {
        "Sid": "IamRolesPoliciesMgmt",
        "Effect": "Allow",
        "Action": [
          "iam:AttachRolePolicy",
          "iam:DetachRolePolicy"
        ],
        "Resource": [
          "arn:aws:iam::*:role/AWS-QuickSetup-*",
          "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
        ],
        "Condition": {
          "ArnEquals": {
            "iam:PolicyARN": [
              "arn:aws:iam::aws:policy/
AWSSystemsManagerEnableExplorerExecutionPolicy",
```

```
            "arn:aws:iam::aws:policy/AWSQuickSetupSSMDeploymentRolePolicy"
          ]
        }
      }
    },
    {
      "Sid": "CfnStackSetsSLR",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/aws-service-role/
stacksets.cloudformation.amazonaws.com/
AWSServiceRoleForCloudFormationStackSetsOrgAdmin",
        "arn:aws:iam::*:role/aws-service-role/ssm.amazonaws.com/
AWSServiceRoleForAmazonSSM",
        "arn:aws:iam::*:role/aws-service-role/
accountdiscovery.ssm.amazonaws.com/AWSServiceRoleForAmazonSSM_AccountDiscovery",
        "arn:aws:iam::*:role/aws-service-role/ssm-quicksetup.amazonaws.com/
AWSServiceRoleForSSMQuickSetup",
        "arn:aws:iam::*:role/aws-service-role/resource-explorer-2.amazonaws.com/
AWSServiceRoleForResourceExplorer"
      ]
    }
  ]
}
EOF

# Create the IAM policy
POLICY_OUTPUT=$(log_cmd "aws iam create-policy --policy-name SSMOnboardingPolicy-
$UNIQUE_ID --policy-document file://ssm-onboarding-policy.json --output json")
POLICY_STATUS=$?
check_error "$POLICY_OUTPUT" $POLICY_STATUS "Failed to create IAM policy"

# Extract the policy ARN
POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:policy/
SSMOnboardingPolicy-[a-f0-9]*')
if [[ -z "$POLICY_ARN" ]]; then
    echo "ERROR: Failed to extract policy ARN" | tee -a "$LOG_FILE"
    exit 1
fi

# Track the created policy
```

```
track_resource "IAM_POLICY" "$POLICY_ARN"
aws iam tag-policy --policy-arn "$POLICY_ARN" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created policy: $POLICY_ARN" | tee -a "$LOG_FILE"

# Step 2: Create and configure IAM role for Systems Manager
echo ""
echo "Step 2: Creating IAM role for Systems Manager..."

# Get current user name
USER_OUTPUT=$(log_cmd "aws sts get-caller-identity --output json")
USER_STATUS=$?
check_error "$USER_OUTPUT" $USER_STATUS "Failed to get caller identity"

# Extract account ID
ACCOUNT_ID=$(echo "$USER_OUTPUT" | grep -o '"Account": "[0-9]*"' | cut -d'"' -f4)
if [[ -z "$ACCOUNT_ID" ]]; then
    echo "ERROR: Failed to extract account ID" | tee -a "$LOG_FILE"
    exit 1
fi

# Generate a unique role name
ROLE_NAME="SSMTutorialRole-$(openssl rand -hex 4)"

# Create trust policy for the role - FIXED: Added cloudformation.amazonaws.com
cat > trust-policy.json << 'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com",
                    "ssm-quicksetup.amazonaws.com",
                    "cloudformation.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        },
        {
            "Effect": "Allow",
            "Principal": {
```

```
                "AWS": "arn:aws:iam::ACCOUNT_ID:root"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

# Replace ACCOUNT_ID placeholder in trust policy
sed -i "s/ACCOUNT_ID/$ACCOUNT_ID/g" trust-policy.json

# Create the IAM role
ROLE_OUTPUT=$(log_cmd "aws iam create-role --role-name $ROLE_NAME --assume-role-
policy-document file://trust-policy.json --description 'Role for Systems Manager
 tutorial' --output json")
ROLE_STATUS=$?
check_error "$ROLE_OUTPUT" $ROLE_STATUS "Failed to create IAM role"

# Extract the role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:role/[^"]*')
if [[ -z "$ROLE_ARN" ]]; then
    echo "ERROR: Failed to extract role ARN" | tee -a "$LOG_FILE"
    cleanup_on_error
    exit 1
fi

# Track the created role
track_resource "IAM_ROLE" "$ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created IAM role: $ROLE_NAME" | tee -a "$LOG_FILE"
echo "Role ARN: $ROLE_ARN" | tee -a "$LOG_FILE"

# Set identity variables for cleanup
IDENTITY_TYPE="role"
IDENTITY_NAME="$ROLE_NAME"

# Attach the policy to the role
ATTACH_OUTPUT=$(log_cmd "aws iam attach-role-policy --role-name $ROLE_NAME --
policy-arn $POLICY_ARN")
ATTACH_STATUS=$?
check_error "$ATTACH_OUTPUT" $ATTACH_STATUS "Failed to attach policy to role
 $ROLE_NAME"
```

```
echo "Policy attached to role: $ROLE_NAME" | tee -a "$LOG_FILE"

# Step 3: Create Systems Manager configuration using Host Management
echo ""
echo "Step 3: Creating Systems Manager configuration..."

# Generate a random identifier for the configuration name
CONFIG_NAME="SSMSetup-$(openssl rand -hex 4)"

# Create configuration file for Systems Manager setup using Host Management
# Added both required parameters for single account deployment based on
 CloudFormation documentation
cat > ssm-config.json << EOF
[
  {
    "Type": "AWSQuickSetupType-SSMHostMgmt",
    "LocalDeploymentAdministrationRoleArn": "$ROLE_ARN",
    "LocalDeploymentExecutionRoleName": "$ROLE_NAME",
    "Parameters": {
      "TargetAccounts": "$ACCOUNT_ID",
      "TargetRegions": "$CURRENT_REGION"
    }
  }
]
EOF

echo "Configuration file created:" | tee -a "$LOG_FILE"
cat ssm-config.json | tee -a "$LOG_FILE"

# Create the configuration manager
CONFIG_OUTPUT=$(log_cmd "aws ssm-quicksetup create-configuration-manager --name
 \"$CONFIG_NAME\" --configuration-definitions file://ssm-config.json --region
 $CURRENT_REGION")
CONFIG_STATUS=$?
check_error "$CONFIG_OUTPUT" $CONFIG_STATUS "Failed to create Systems Manager
 configuration"

# Extract the manager ARN
MANAGER_ARN=$(echo "$CONFIG_OUTPUT" | grep -o 'arn:aws:ssm-quicksetup:[^"]*')
if [[ -z "$MANAGER_ARN" ]]; then
    echo "ERROR: Failed to extract manager ARN" | tee -a "$LOG_FILE"
    exit 1
fi
```

```
# Track the created configuration manager
track_resource "SSM_CONFIG_MANAGER" "$MANAGER_ARN"

echo "Created Systems Manager configuration: $MANAGER_ARN" | tee -a "$LOG_FILE"

# Step 4: Verify the setup
echo ""
echo "Step 4: Verifying the setup..."

# Wait for the configuration to be fully deployed
echo "Waiting for the configuration to be deployed (this may take a few
 minutes)..."
sleep 30

# Check the configuration manager status
VERIFY_OUTPUT=$(log_cmd "aws ssm-quicksetup get-configuration-manager --manager-
arn $MANAGER_ARN --region $CURRENT_REGION")
VERIFY_STATUS=$?
check_error "$VERIFY_OUTPUT" $VERIFY_STATUS "Failed to verify configuration
 manager"

echo "Systems Manager setup completed successfully!" | tee -a "$LOG_FILE"

# List the created resources
echo ""
echo "==========================================="
echo "CREATED RESOURCES"
echo "==========================================="
for resource in "${CREATED_RESOURCES[@]+"${CREATED_RESOURCES[@]}"}"; do
    echo "$resource"
done

# Prompt for cleanup
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE="y"

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..." | tee -a "$LOG_FILE"
    cleanup_resources
```

```
    echo "Cleanup completed." | tee -a "$LOG_FILE"
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later." | tee -a "$LOG_FILE"
fi

echo ""
echo "Script execution completed. See $LOG_FILE for details."

# Clean up temporary files
rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/null ||
 true
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des commandes de l'AWS CLI .
  - AttachRolePolicy
  - CreateConfigurationManager
  - CreatePolicy
  - CreateRole
  - DeleteConfigurationManager
  - DeletePolicy
  - DeleteRole
  - DetachRolePolicy
  - GetCallerIdentity
  - GetConfigurationManager

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezL'utilisation de ce service avec un AWS Kit SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Sécurité dans IAM et AWS STS

La sécurité du cloud AWS est la priorité absolue. En tant que AWS client, vous bénéficiez de centres de données et d'architectures réseau conçus pour répondre aux exigences des entreprises les plus sensibles en matière de sécurité.

La sécurité est une responsabilité partagée entre vous AWS et vous. Le modèle de responsabilité partagée décrit ceci comme la sécurité du cloud et la sécurité dans le cloud :

- Sécurité du cloud : AWS est chargée de protéger l'infrastructure qui exécute les AWS services dans le AWS cloud. AWS vous fournit également des services que vous pouvez utiliser en toute sécurité. Des auditeurs tiers testent et vérifient régulièrement l'efficacité de notre sécurité dans le cadre des programmes de AWS conformité Programmes de de conformité. Pour en savoir plus sur les programmes de conformité qui s'appliquent à Gestion des identités et des accès AWS (IAM), voir AWS Services concernés par programme de conformitéAWS .
- Sécurité dans le cloud — Votre responsabilité est déterminée par le AWS service que vous utilisez. Vous êtes également responsable d'autres facteurs, y compris la sensibilité de vos données, les exigences de votre entreprise, ainsi que la législation et la réglementation applicables.

Cette documentation vous aide à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation de Gestion des identités et des accès AWS (IAM) et AWS Security Token Service (AWS STS). Les rubriques suivantes vous montrent comment configurer l'IAM et comment AWS STS atteindre vos objectifs de sécurité et de conformité. Vous apprendrez également à utiliser d'autres AWS services qui vous aident à surveiller et à sécuriser vos ressources IAM.

Table des matières

- AWS informations d'identification de sécurité
- AWS directives relatives aux audits de sécurité
- Protection des données dans Gestion des identités et des accès AWS
- Connexion et surveillance Gestion des identités et des accès AWS
- Validation de conformité pour Gestion des identités et des accès AWS
- Résilience dans Gestion des identités et des accès AWS
- Sécurité de l'infrastructure dans Gestion des identités et des accès AWS
- Analyse de la configuration et des vulnérabilités dans Gestion des identités et des accès AWS

- AWS politiques gérées pour Gestion des identités et des accès AWS, IAM Access Analyzer et gestionnaire d'accès aux comptes
- Fonctions de sécurité en dehors d'IAM

AWS informations d'identification de sécurité

Lorsque vous interagissez avec AWS, vous spécifiez vos informations de AWS sécurité pour vérifier qui vous êtes et si vous êtes autorisé à accéder aux ressources que vous demandez. AWS utilise les informations d'identification de sécurité pour authentifier et autoriser vos demandes.

Par exemple, si vous souhaitez télécharger un fichier protégé à partir d'un compartiment Amazon Simple Storage Service (Amazon S3), vos informations d'identification doivent autoriser cet accès. Si vos informations d'identification n'indiquent pas que vous êtes autorisé à télécharger le fichier, AWS refuse votre demande. Toutefois, vos informations d'identification de AWS sécurité ne sont pas nécessaires pour télécharger un fichier dans un compartiment Amazon S3 partagé publiquement.

Il existe différents types d'utilisateurs AWS, chacun possédant ses propres identifiants de sécurité :

- Titulaire du compte (utilisateur root) — L'utilisateur qui a créé le Compte AWS et dispose d'un accès complet.
- AWS IAM Identity Center utilisateurs : utilisateurs gérés dans AWS IAM Identity Center.
- Principaux fédérés : utilisateurs de fournisseurs d'identité externes auxquels un accès temporaire est accordé par le AWS biais de la fédération. Pour plus d'informations sur les identités fédérées, consultez Fournisseurs d'identité et fédération au sein AWS.
- Utilisateurs IAM : utilisateurs individuels créés au sein du service Gestion des identités et des accès AWS (IAM).

Les utilisateurs disposent d'informations d'identification de sécurité à long terme ou temporaires. L'utilisateur root, l'utilisateur IAM et les clés d'accès possèdent des informations d'identification de sécurité à long terme qui n'expirent pas. Pour protéger les informations d'identification à long terme, il convient de mettre en place des processus pour gérer les clés d'accès, modifier les mots de passe et activer MFA.

Pour simplifier la gestion des informations d'identification de l'utilisateur root sur l'ensemble des comptes membres AWS Organizations, vous pouvez sécuriser de manière centralisée les informations d'identification de l'utilisateur root de votre compte Comptes AWS géré

AWS Organizations. Gestion centralisée de l'accès root pour les comptes membresvous permet de supprimer de manière centralisée et d'empêcher la restauration à long terme des informations d'identification de l'utilisateur root, empêchant ainsi tout accès root involontaire à grande échelle.

Les rôles IAM et les utilisateurs dans AWS IAM Identity Center utilisateurs principaux AWS STS fédérés disposent d'informations d'identification de sécurité temporaires. Les informations d'identification de sécurité temporaires expirent après une période définie ou lorsque l'utilisateur met fin à sa session. Les informations d'identification temporaires fonctionnent de manière presque identique aux informations d'identification des clés d'accès à long terme, à quelques différences près :

- Les informations d'identification de sécurité temporaires sont à court terme, comme leur nom l'indique. Elles peuvent être configurées pour être valides de quelques minutes à plusieurs heures. Une fois les informations d'identification AWS expirées, il ne les reconnaît plus ou n'autorise aucun type d'accès à partir des demandes d'API effectuées avec elles.
- Les informations d'identification de sécurité temporaires ne sont pas stockées avec l'utilisateur, mais sont générées automatiquement et fournies à l'utilisateur sur demande. Lorsque (ou même avant) les informations d'identification de sécurité temporaires arrivent à expiration, l'utilisateur peut en demander de nouvelles, tant qu'il y est autorisé.

Par conséquent, les informations d'identification temporaires présentent les avantages suivants par rapport aux informations d'identification à long terme :

- Il n'est pas nécessaire de distribuer ou d'intégrer des informations d'identification AWS de sécurité à long terme dans une application.
- Vous pouvez donner accès à vos AWS ressources aux utilisateurs sans avoir à définir leur AWS identité. Les informations d'identification temporaires servent de base aux rôles et à la fédération d'identité.
- Les informations d'identification de sécurité temporaires ont une durée de vie limitée. Il n'est donc pas nécessaire de les mettre à jour ou de les révoquer explicitement lorsqu'elles deviennent obsolètes. Une fois que les informations d'identification de sécurité temporaires arrivent à expiration, elles ne peuvent pas être réutilisées. Vous pouvez spécifier le délai de validité des informations d'identification, jusqu'à une certaine limite.

Considérations sur la sécurité

Nous vous recommandons de tenir compte des informations suivantes lorsque vous déterminez les mesures de sécurité pour votre Compte AWS :

- Lorsque vous créez un Compte AWS, nous créons le compte utilisateur root. Les informations d'identification de l'utilisateur root (propriétaire du compte) permettent un accès complet à toutes les ressources du compte. La première tâche que vous effectuez avec l'utilisateur root consiste à accorder à un autre utilisateur des autorisations administratives Compte AWS afin de minimiser l'utilisation de l'utilisateur root.
- L'authentification multi-facteurs (Multi-Factor Authentication, MFA) fournit un niveau de sécurité supplémentaire pour les utilisateurs pouvant accéder à votre Compte AWS. Pour plus de sécurité, nous vous recommandons d'exiger l'authentification MFA sur les Utilisateur racine d'un compte AWS informations d'identification et sur tous les utilisateurs IAM. Pour de plus amples informations, veuillez consulter AWS Multi-factor authentification dans IAM.
- AWS nécessite différents types d'identifiants de sécurité, en fonction de votre mode d'accès AWS et du type d' AWS utilisateur que vous êtes. Par exemple, vous utilisez les informations de connexion Console de gestion AWS pendant que vous utilisez les touches d'accès pour effectuer des appels programmatiques à. AWS Pour obtenir de l'aide pour déterminer votre type d'utilisateur et votre page de connexion, consultez la section Qu'est-ce que la AWS connexion dans le guide de l'Connexion à AWS utilisateur.
- Vous ne pouvez pas utiliser des politiques IAM pour refuser l'accès aux ressources de manière explicite à l'utilisateur root. Vous ne pouvez utiliser une politique AWS Organizations de contrôle des services (SCP) que pour limiter les autorisations de l'utilisateur root.
- Si vous oubliez ou perdez votre mot de passe root, vous devez avoir accès à l'adresse e-mail associée à votre compte pour le réinitialiser.
- Si vous perdez vos clés d'accès de l'utilisateur root, vous devez vous connecter à votre compte en tant qu'utilisateur root pour en créer de nouvelles.
- N'utilisez pas l'utilisateur root de vos tâches courantes. Optez pour effectuer les tâches que seul l'utilisateur root peut effectuer. Pour obtenir la liste complète des tâches qui nécessitent que vous vous connectiez en tant qu'utilisateur root, veuillez consulter Tâches nécessitant les informations d'identification de l'utilisateur root.
- Les informations d'identification de sécurité sont spécifiques au compte. Si vous avez accès à plusieurs Comptes AWS, vous disposez d'informations d'identification distinctes pour chaque compte.

- Les politiques déterminent les actions qu'un utilisateur, un rôle ou un membre d'un groupe d'utilisateurs peut effectuer, sur quelles AWS ressources et dans quelles conditions. À l'aide de politiques, vous pouvez contrôler en toute sécurité l'accès Services AWS et les ressources de votre Compte AWS. Si vous devez modifier ou révoquer des autorisations en réponse à un événement de sécurité, vous supprimez ou modifiez les politiques au lieu de modifier directement l'identité.
- Veillez à enregistrer les informations de connexion de votre utilisateur IAM Emergency Access et toutes les clés d'accès que vous avez créées pour un accès programmatique dans un emplacement sécurisé. Si vous perdez vos clés d'accès, vous devez vous connecter à votre compte pour en créer de nouvelles.
- Nous vous recommandons vivement d'utiliser les informations d'identification temporaires fournies par les rôles IAM et les principaux fédérés plutôt que les informations d'identification à long terme fournies par les utilisateurs IAM et les clés d'accès.

Accès programmatique avec AWS Informations d'identification de sécurité

Nous vous recommandons d'utiliser des clés d'accès de courte durée dans la mesure du possible pour effectuer des appels programmatiques vers AWS ou pour utiliser le AWS Command Line Interface ou Outils AWS pour PowerShell. Cependant, vous pouvez également utiliser des clés AWS d'accès à long terme à ces fins.

Lorsque vous créez une clé d'accès à long terme, vous générez l'ID de clé d'accès (par exemple, `AKIAIOSFODNN7EXAMPLE`) et la clé d'accès secrète (par exemple, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`) sous la forme d'un ensemble. La clé d'accès secrète est accessible en téléchargement uniquement au moment de sa création. Si vous ne téléchargez pas votre clé d'accès secrète ou si vous la perdez, vous devrez la recréer.

Dans de nombreux cas, vous n'avez pas besoin de clés d'accès à long terme qui n'expirent jamais (comme lorsque vous créez des clés d'accès pour un utilisateur IAM). Au lieu de cela, vous pouvez créer des rôles IAM et générer des informations d'identification de sécurité temporaires. Les informations d'identification de sécurité temporaires comprennent non seulement un ID de clé d'accès et une clé d'accès secrète, mais également un jeton de sécurité qui indique la date d'expiration des informations d'identification. Après leur expiration, elles ne sont plus valides. Pour de plus amples informations, consultez Alternatives aux clés d'accès à long terme.

Les ID de clé d'accès commençant par `AKIA` sont des clés d'accès à long terme pour un utilisateur IAM ou un utilisateur Compte AWS root. Les ID de clé d'accès commençant par `ASIA` sont des clés d'accès temporaires que vous créez à l'aide d' AWS STS opérations.

Les utilisateurs ont besoin d'un accès programmatique s'ils souhaitent interagir avec des AWS personnes extérieures au Console de gestion AWS. La manière d'accorder l'accès par programmation dépend du type d'utilisateur qui accède AWS.

Pour accorder aux utilisateurs un accès programmatique, choisissez l'une des options suivantes.

Quel utilisateur a besoin d'un accès programmatique ?	À	Méthode
IAM	(Recommandé) Utilisez les informations d'identification de la console comme informations d'identification temporaires pour signer les AWS CLI demandes programmatiques adressées aux AWS SDK ou AWS aux API.	Suivez les instructions de l'interface que vous souhaitez utiliser. • Pour cela AWS CLI, consultez la section Connexion pour le développement AWS local dans le Guide de AWS Command Line Interface l'utilisateur. • Pour les AWS kits de développement, consultez la section Connexion pour le développement AWS local dans le Guide de référence AWS des kits de développement logiciel et des outils.
Identité de la main-d'œuvre (Utilisateurs gérés dans IAM Identity Center)	Utilisez des informations d'identification temporaires pour signer les demandes programmatiques adressées aux AWS CLI AWS SDK ou AWS aux API.	Suivez les instructions de l'interface que vous souhaitez utiliser. • Pour cela AWS CLI, reportez-vous AWS CLI à la section Configuration de l'utilisation AWS IAM Identity Center dans le

Quel utilisateur a besoin d'un accès programmatique ?	À	Méthode
		Guide de AWS Command Line Interface l'utilisateur. • Pour les AWS kits de développement logiciel, les outils et les AWS API, consultez la section Authentification IAM Identity Center dans le Guide de référence AWS des kits de développement logiciel et des outils.
IAM	Utilisez des informations d'identification temporaires pour signer les demandes programmatiques adressées aux AWS CLI AWS SDK ou AWS aux API.	Suivez les instructions de la section Utilisation des informations d'identification temporaires avec AWS les ressources du Guide de l'utilisateur IAM.

Quel utilisateur a besoin d'un accès programmatique ?	À	Méthode
IAM	(Non recommandé) Utilisez des informations d'identification à long terme pour signer des demandes programmatiques adressées aux AWS CLI AWS SDK ou AWS aux API.	Suivez les instructions de l'interface que vous souhaitez utiliser. • Pour cela AWS CLI, consultez la section Authentification à l'aide des informations d'identification utilisateur IAM dans le Guide de l'AWS Command Line Interface utilisateur. • Pour les AWS kits de développement logiciel et les outils, consultez la section Authentification à l'aide d'informations d'identification à long terme dans le Guide de référence AWS des kits de développement logiciel et des outils. • Pour les AWS API, consultez la section Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur IAM.

Alternatives aux clés d'accès à long terme

Pour de nombreux cas d'utilisation courants, il existe des alternatives aux clés d'accès à long terme. Pour améliorer la sécurité de votre compte, tenez compte des points suivants.

- N'intégrez pas de clés d'accès à long terme et de clés d'accès secrètes dans le code de votre application ou dans un référentiel de code. Utilisez AWS Secrets Manager plutôt une autre solution de gestion des secrets, afin de ne pas avoir à coder les clés en dur en texte clair. L'application ou le client peut ensuite récupérer des secrets en cas de besoin. Pour plus d'informations, voir Qu'est-ce que c'est AWS Secrets Manager ? dans le Guide de AWS Secrets Manager l'utilisateur.

- Utilisez les rôles IAM pour générer des informations d'identification de sécurité temporaires dans la mesure du possible : utilisez toujours des mécanismes permettant de délivrer des identifiants de sécurité temporaires, lorsque cela est possible, plutôt que des clés d'accès à long terme. Les informations d'identification de sécurité temporaires sont plus sécurisées car elles ne sont pas stockées avec l'utilisateur, mais sont générées automatiquement et fournies à l'utilisateur sur demande. Les informations d'identification de sécurité temporaires ont une durée de vie limitée, ce qui vous évite d'avoir à les gérer ou à les mettre à jour. Les mécanismes qui fournissent des clés d'accès temporaires incluent les rôles IAM ou l'authentification d'un utilisateur d'IAM Identity Center. Pour les machines qui s'exécutent en dehors de AWS vous, vous pouvez utiliser Gestion des identités et des accès AWS Roles Anywhere.
- Utilisez des alternatives aux clés d'accès à long terme pour l' AWS Command Line Interface (AWS CLI) ou **`aws-shell`** — Les alternatives incluent ce qui suit.
  - Connectez-vous pour le développement AWS local à l'aide des informations d'identification de la console. Vous pouvez utiliser la AWS CLI version 2 et la `aws login` commande pour générer des informations d'identification à court terme afin d'exécuter des AWS CLI commandes à l'aide des informations d'identification de votre console. Pour en savoir plus, consultez la section Connexion pour le développement AWS local dans le Guide de AWS Command Line Interface l'utilisateur.
  - AWS CloudShellest un shell pré-authentifié basé sur un navigateur que vous pouvez lancer directement depuis. Console de gestion AWS Vous pouvez exécuter AWS CLI des commandes Services AWS via votre shell préféré (Bash, Powershell ou Z shell). Dans ce cas, il n'est pas nécessaire de télécharger ou d'installer des outils de ligne de commande. Pour plus d'informations, consultez Présentation de AWS CloudShell dans le Guide de l'utilisateur AWS CloudShell .
- Ne créez pas de clés d'accès à long terme pour les utilisateurs humains qui ont besoin d'accéder à des applications. Sinon Services AWS , IAM Identity Center peut générer des informations d'accès temporaires pour les utilisateurs de votre IdP externe. Services AWS Il n'est donc plus nécessaire de créer et de gérer des informations d'identification à long terme dans IAM. Dans IAM Identity Center, créez un ensemble d'autorisations IAM Identity Center qui accorde l'accès aux utilisateurs

IdP externes. Attribuez ensuite à un groupe depuis IAM Identity Center l'autorisation définie dans la zone sélectionnée Comptes AWS. Pour plus d'informations, consultez Qu'est-ce que AWS IAM Identity Center, Connexion à votre fournisseur d'identité externe et Ensembles d'autorisations dans le Guide de AWS IAM Identity Center l'utilisateur.

- Ne stockez pas les clés d'accès à long terme dans un service de AWS calcul. Attribuez plutôt un rôle IAM aux ressources de calcul. Cela fournit automatiquement des informations d'identification temporaires pour accorder l'accès. Par exemple, lorsque vous créez un profil d'instance attaché à une instance Amazon EC2, vous pouvez attribuer un AWS rôle à l'instance et la mettre à la disposition de toutes ses applications. Un profil d'instance contient le rôle et permet aux programmes qui s'exécutent sur l'instance Amazon EC2 d'obtenir des informations d'identification temporaires. Pour en savoir plus, consultez Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2.

AWS directives relatives aux audits de sécurité

Effectuez régulièrement un audit de votre configuration de sécurité pour vérifier qu'elle répond aux besoins actuels de votre entreprise. Un audit vous donne la possibilité de supprimer les utilisateurs, les rôles, les groupes et les politiques IAM inutiles, et de vérifier que vos utilisateurs et vos logiciels ne disposent pas d'autorisations excessives.

Vous trouverez ci-dessous des directives pour examiner et surveiller systématiquement vos AWS ressources afin de respecter les meilleures pratiques en matière de sécurité.

> **Tip**
>
> Vous pouvez surveillez votre utilisation d'IAM, conformément aux bonnes pratiques de sécurité, à l'aide d'AWS Security Hub CSPM. Security Hub CSPM utilise des contrôles de sécurité pour évaluer les configurations des ressources et les normes de sécurité afin de vous aider à vous conformer aux différents cadres de conformité. Pour plus d'informations sur l'utilisation de Security Hub CSPM pour évaluer les ressources IAM, consultez la section Contrôles AWS Identity and Access Management dans le guide de l' AWS Security Hub CSPM utilisateur.

Table des matières

- Quand effectuer un audit de sécurité ?
- Consignes pour l'audit

- Vérifiez les informations d'identification AWS de votre compte
- Examen de vos utilisateurs IAM
- Examen de vos groupes IAM
- Examen de vos rôles IAM
- Examen de vos fournisseurs IAM pour SAML et OpenID Connect (OIDC)
- Examen de vos applications mobiles
- Conseils pour l'examen des politiques IAM

Quand effectuer un audit de sécurité ?

Effectuez un audit de votre configuration de sécurité dans les cas suivants :

- De manière régulière. Exécutez les étapes décrites dans ce document à intervalles réguliers, à titre de bonne pratique de sécurité.
- Si des changements se produisent dans votre organisation, par exemple lorsque des personnes quittent l'organisation.
- Si vous avez cessé d'utiliser un ou plusieurs AWS services individuels pour vérifier que vous avez supprimé les autorisations dont les utilisateurs de votre compte n'ont plus besoin.
- Si vous avez ajouté ou supprimé des logiciels dans vos comptes, tels que des applications sur des instances Amazon EC2, des OpsWorks piles, des CloudFormation modèles, etc.
- Si vous pensez qu'une personne non autorisée a pu avoir accès à votre compte.

Consignes pour l'audit

Lorsque vous examinez la configuration de sécurité de votre compte, suivez ces consignes :

- Soyez minutieux. Regardez tous les aspects de votre configuration de sécurité, y compris ceux qui sont rarement utilisés.
- Ne faites pas de suppositions. Si vous ne connaissez pas vraiment certains aspects de votre configuration de sécurité (par exemple, le raisonnement qui justifie une stratégie particulière ou l'existence d'un rôle), étudiez les besoins de l'entreprise jusqu'à ce que vous compreniez le risque potentiel.
- Simplifiez les choses. Pour faciliter l'audit (et la gestion), utilisez les groupes et les rôles IAM, les schémas d'affectation de nom cohérents et les stratégies simples.

Vérifiez les informations d'identification AWS de votre compte

Procédez comme suit lorsque vous auditez les informations d'identification de votre AWS compte :

1. Si vous avez des clés d'accès pour votre utilisateur root que vous n'utilisez pas, vous pouvez les supprimer. Nous vous recommandons vivement de ne pas utiliser les clés d'accès root dans le cadre de votre travail quotidien AWS, mais plutôt d'utiliser des utilisateurs dotés d'informations d'identification temporaires, telles que utilisateurs dans AWS IAM Identity Center.
2. Si vous avez besoin de clés d'accès de votre compte, assurez-vous de les mettre à jour en cas de besoin.

Examen de vos utilisateurs IAM

Exécutez les étapes suivantes lorsque vous effectuez un audit de vos utilisateurs IAM existants :

1. Répertoriez vos utilisateurs, puis supprimez les utilisateurs qui ne sont pas nécessaires.
2. Supprimez les utilisateurs des groupes auxquels ils n'ont pas besoin d'accéder.
3. Examinez les stratégies associées aux groupes dans lesquels se trouve l'utilisateur. Consultez Conseils pour l'examen des politiques IAM.
4. Supprimez les informations d'identification de sécurité inutiles pour l'utilisateur ou susceptibles d'avoir été exposées. Par exemple, un utilisateur IAM utilisé pour une application n'a pas besoin de mot de passe (nécessaire uniquement pour se connecter à des AWS sites Web). De même, si un utilisateur n'utilise pas de clés d'accès, il n'y a aucune raison qu'il en ait une. Pour plus d'informations, consultez Gestion des mots de passe des utilisateurs IAM et Gestion des clés d'accès pour les utilisateurs IAM.

   Vous pouvez générer et télécharger un rapport sur les informations d'identification qui répertorie tous les utilisateurs IAM de votre compte et le statut de leurs diverses informations d'identification, notamment leurs mots de passe, clés d'accès et dispositifs MFA. Pour les mots de passe et les clés d'accès, le rapport d'informations d'identification indique la date et l'heure auxquelles le mot de passe ou la clé d'accès a été utilisé(e) pour la dernière fois. Envisagez de supprimer de votre compte les informations d'identification qui n'ont pas été utilisées récemment. (Ne supprimez pas votre utilisateur d'accès d'urgence.) Pour plus d'informations, consultez la section Obtenir des rapports d'identification pour votre AWS compte.
5. Mettez à jour les mots de passe et les clés d'accès lorsque cela est nécessaire pour les cas d'utilisation nécessitant des informations d'identification à long terme. Pour plus d'informations,

consultez Gestion des mots de passe des utilisateurs IAM et Gestion des clés d'accès pour les utilisateurs IAM.

6. Il est recommandé d'obliger les utilisateurs humains à utiliser la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires. Si possible, passez du statut d'utilisateurs IAM à celui de principaux fédérés, tels que les utilisateurs d'IAM Identity Center. Conservez le nombre minimum d'utilisateurs IAM requis pour vos applications.

Examen de vos groupes IAM

Exécutez les étapes suivantes lorsque vous effectuez un audit de vos groupes IAM :

1. Répertoriez vos groupes, puis supprimez les groupes que vous n'utilisez pas.
2. Examinez les utilisateurs de chaque groupe et supprimez les utilisateurs qui n'y appartiennent pas.
3. Examinez les stratégies associées au groupe. Consultez Conseils pour l'examen des politiques IAM.

Examen de vos rôles IAM

Exécutez les étapes suivantes lorsque vous effectuez un audit de vos rôles IAM :

1. Répertoriez vos rôles, puis supprimez les rôles que vous n'utilisez pas.
2. Examinez la stratégie d'approbation du rôle. Assurez-vous de connaître le principal et de comprendre pourquoi ce compte ou cet utilisateur doit être capable d'assumer le rôle.
3. Examinez la stratégie d'accès du rôle pour vous assurer qu'elle accorde les autorisations appropriées à la personne qui assume le rôle (voir Conseils pour l'examen des politiques IAM).

Examen de vos fournisseurs IAM pour SAML et OpenID Connect (OIDC)

Si vous avez créé une entité IAM pour établir une relation de confiance avec un fournisseur d'identité (IdP) SAML ou OIDC, procédez comme suit :

1. Supprimez les fournisseurs inutilisés.
2. Téléchargez et consultez les documents de AWS métadonnées pour chaque IdP SAML et assurez-vous qu'ils reflètent les besoins actuels de votre entreprise.

3. Obtenez les derniers documents de métadonnées auprès du SAML IdPs et mettez à jour le fournisseur dans IAM.

Examen de vos applications mobiles

Si vous avez créé une application mobile qui envoie des demandes à AWS, procédez comme suit :

1. Vérifiez que l'application mobile ne contient pas de clés d'accès incorporées, même si elles se trouvent dans le stockage chiffré.
2. Obtenez des informations d'identification temporaires pour l'application en utilisant un APIs outil conçu à cet effet.

> **Note**
>
> Nous vous recommandons d'utiliser Amazon Cognito pour gérer l'identité de l'utilisateur dans votre application. Ce service vous permet d'authentifier les utilisateurs à l'aide de Login with Amazon, Facebook, Google ou tout autre fournisseur d'identité compatible avec OpenID Connect (OIDC). Pour plus d'informations sur les pools d'identité Amazon Cognito, consultez le Manuel du développeur Amazon Cognito

Conseils pour l'examen des politiques IAM

Les politiques sont puissantes et subtiles, il est donc important d'étudier et de comprendre les autorisations octroyées par chaque politique. Utilisez les consignes suivantes lors de l'examen des stratégies :

- Attachez des politiques à des groupes ou à des rôles plutôt qu'à des utilisateurs individuels. Si un utilisateur individuel dispose d'une stratégie, assurez-vous de comprendre pourquoi cet utilisateur a besoin d'une stratégie.
- Assurez-vous que les utilisateurs, les groupes d'utilisateurs et les rôles IAM disposent des autorisations dont ils ont besoin et n'ont aucune autorisation supplémentaire.
- Utilisez le simulateur de politiques IAM pour tester les politiques associées à des utilisateurs ou des groupes.
- N'oubliez pas que les autorisations d'un utilisateur sont le résultat de toutes les politiques applicables, à la fois des politiques basées sur l'identité (sur les utilisateurs, les groupes ou les

rôles) et des politiques basées sur les ressources (sur des ressources telles que les compartiments Amazon S3, les files d'attente Amazon SQS, les rubriques Amazon SNS et les clés). AWS KMS Il est important d'examiner toutes les stratégies qui s'appliquent à un utilisateur et de comprendre l'ensemble complet des autorisations octroyées à un utilisateur individuel.

- Sachez qu'en autorisant un utilisateur à créer un utilisateur, un groupe, un rôle ou une politique IAM et à associer une politique à l'entité principale, toutes les autorisations à toutes les ressources de votre compte lui sont en fait octroyées. Les utilisateurs qui peuvent créer des politiques et les associer à un utilisateur, un groupe ou un rôle peuvent octroyer eux-mêmes n'importe quelle autorisation. En général, n'octroyez pas d'autorisations IAM à des utilisateurs ou des rôles auxquels vous n'approuvez pas l'accès complet aux ressources de votre compte. Lorsque vous effectuez votre audit de sécurité, confirmez que les autorisations IAM suivantes sont octroyées à des identités de confiance :
  - `iam:PutGroupPolicy`
  - `iam:PutRolePolicy`
  - `iam:PutUserPolicy`
  - `iam:CreatePolicy`
  - `iam:CreatePolicyVersion`
  - `iam:AttachGroupPolicy`
  - `iam:AttachRolePolicy`
  - `iam:AttachUserPolicy`
- Assurez-vous que les stratégies n'octroient pas d'autorisations aux services que vous n'utilisez pas. Par exemple, si vous utilisez des politiques AWS gérées, assurez-vous que les politiques AWS gérées utilisées dans votre compte concernent les services que vous utilisez réellement. Pour savoir quelles politiques AWS gérées sont utilisées dans votre compte, utilisez l'`GetAccountAuthorizationDetails`API IAM (AWS CLI commande : `aws iam get-account-authorization-details`).
- Si la politique octroie à un utilisateur l'autorisation de lancer une instance Amazon EC2, elle peut également autoriser l'action `iam:PassRole`, mais si c'est le cas, elle doit explicitement répertorier les rôles que l'utilisateur peut transmettre à l'instance Amazon EC2.
- Examinez toutes les valeurs de l'élément `Action` ou `Resource` qui incluent les `*`. Dans la mesure du possible, octroyez l'accès `Allow` aux actions et aux ressources individuelles dont les utilisateurs ont besoin. Toutefois, voici les raisons pour lesquelles il peut être approprié d'utiliser `*` dans une stratégie :
  - La politique est conçue pour octroyer des autorisations au niveau administratif.

- Le caractère générique est utilisé pour un ensemble d'actions similaires (par exemple, `Describe*`) pour vous faciliter les choses, et vous vous sentez à l'aise avec la liste complète des actions qui sont référencées de cette façon.
- Le caractère générique est utilisé pour indiquer une classe de ressources ou un chemin d'accès aux ressources (par exemple, `arn:aws:iam::account-id:users/division_abc/*`), et vous vous sentez à l'aise pour octroyer l'accès à toutes les ressources dans cette classe ou ce chemin d'accès.
- Une action de service ne prend pas en charge les autorisations au niveau des ressources, et le seul choix pour une ressource est `*`.

- Examinez les noms de stratégie pour vérifier qu'ils reflètent la fonction de la stratégie. Par exemple, même si une stratégie peut avoir un nom qui inclut « lecture seule », la stratégie peut en fait octroyer des autorisations de lecture ou de modification.

Pour de plus amples informations sur la planification de votre audit de sécurité, consultez Meilleures pratiques de sécurité, d'identité et de conformité dans le Centre d'architecture AWS .

Protection des données dans Gestion des identités et des accès AWS

Le modèle de responsabilité AWS partagée Le modèle de s'applique à la protection des données dans Gestion des identités et des accès AWS. Comme décrit dans ce modèle, AWS est chargé de protéger l'infrastructure mondiale qui gère tous les AWS Cloud. La gestion du contrôle de votre contenu hébergé sur cette infrastructure relève de votre responsabilité. Vous êtes également responsable des tâches de configuration et de gestion de la sécurité des Services AWS que vous utilisez. Pour plus d'informations sur la confidentialité des données, consultez la FAQ sur la confidentialité des données et les . Pour plus d'informations sur la protection des données en Europe, consultez le Centre du règlement général sur la protection des données (RGPD).

À des fins de protection des données, nous vous recommandons de protéger les Compte AWS informations d'identification et de configurer les utilisateurs individuels avec AWS IAM Identity Center ou Gestion des identités et des accès AWS (IAM). Ainsi, chaque utilisateur se voit attribuer uniquement les autorisations nécessaires pour exécuter ses tâches. Nous vous recommandons également de sécuriser vos données comme indiqué ci-dessous :

- Utilisez l'authentification multifactorielle (MFA) avec chaque compte.

- SSL/TLS À utiliser pour communiquer avec AWS les ressources. Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Configurez l'API et la journalisation de l'activité des utilisateurs avec AWS CloudTrail. Pour plus d'informations sur l'utilisation des CloudTrail sentiers pour capturer AWS des activités, consultez la section Utilisation des CloudTrail sentiers dans le guide de AWS CloudTrail l'utilisateur.
- Utilisez des solutions de AWS chiffrement, ainsi que tous les contrôles de sécurité par défaut qu'ils contiennent Services AWS.
- Utilisez des services de sécurité gérés avancés tels qu'Amazon Macie, qui contribuent à la découverte et à la sécurisation des données sensibles stockées dans Amazon S3.
- Si vous avez besoin de modules cryptographiques validés par la norme FIPS 140-3 pour accéder AWS via une interface de ligne de commande ou une API, utilisez un point de terminaison FIPS. Pour plus d'informations sur les points de terminaison FIPS disponibles, consultez Norme FIPS (Federal Information Processing Standard) 140-3.

Nous vous recommandons fortement de ne jamais placer d'informations confidentielles ou sensibles, telles que les adresses e-mail de vos clients, dans des balises ou des champs de texte libre tels que le champ Nom. Cela inclut lorsque vous travaillez avec IAM ou un autre utilisateur à Services AWS l'aide de la console, de l'API ou des AWS SDK. AWS CLI Toutes les données que vous entrez dans des balises ou des champs de texte de forme libre utilisés pour les noms peuvent être utilisées à des fins de facturation ou dans les journaux de diagnostic. Si vous fournissez une adresse URL à un serveur externe, nous vous recommandons fortement de ne pas inclure d'informations d'identification dans l'adresse URL permettant de valider votre demande adressée à ce serveur.

Chiffrement des données dans IAM et AWS STS

Le chiffrement des données se divise généralement en deux catégories : le chiffrement au repos et le chiffrement en transit.

Chiffrement au repos

Les données collectées et stockées par IAM sont chiffrées au repos.

- IAM : les données collectées et stockées dans IAM comprennent les adresses IP, les métadonnées du compte client et les données d'identification des clients, y compris les mots de passe. Les métadonnées du compte client et les données d'identification des clients sont chiffrées au repos à l'aide d'AES 256 ou sont hachés à l'aide de SHA 256.

- AWS STS— AWS STS ne collecte pas le contenu des clients, à l'exception des journaux de service qui enregistrent les demandes réussies, erronées ou erronées adressées au service.

Chiffrement en transit

Les données d'identification des clients, y compris les mots de passe, sont chiffrées en transit à l'aide des protocoles TLS 1.2 et 1.3. Tous les AWS STS terminaux prennent en charge le protocole HTTPS pour le chiffrement des données en transit. Pour obtenir la liste des AWS STS points de terminaison, consultezAWS STS Régions et points de terminaison.

Gestion des clés dans IAM et AWS STS

Vous ne pouvez pas gérer les clés de chiffrement à l'aide d'IAM ou AWS STS. Pour plus d'informations sur les clés de chiffrement, voir Qu'est-ce que c'est AWS KMS ? dans le guide AWS Key Management Service du développeur

Confidentialité du trafic interréseau dans IAM et AWS STS

Les demandes IAM doivent être effectuées à l'aide du protocole TLS (Transport Layer Security). Vous pouvez sécuriser les connexions au AWS STS service en utilisant des points de terminaison VPC. Pour en savoir plus, consultez Points de terminaison de VPC d'Interface.

Connexion et surveillance Gestion des identités et des accès AWS

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances de Gestion des identités et des accès AWS (IAM), AWS Security Token Service (AWS STS) et de vos autres AWS solutions. AWS fournit plusieurs outils pour surveiller vos AWS ressources et répondre aux incidents potentiels :

- AWS CloudTrailcapture tous les appels d'API pour IAM et AWS STS en tant qu'événements, y compris les appels depuis la console et les appels d'API. Pour en savoir plus sur l'utilisation CloudTrail avec IAM et AWS STS, voirJournalisation IAM et AWS STS Appels d'API avec AWS CloudTrail. Pour plus d'informations CloudTrail, consultez le guide de AWS CloudTrail l'utilisateur.
- Gestion des identités et des accès AWS et Access Analyzer vous aide à identifier les ressources de votre organisation et les comptes, tels que les compartiments Amazon S3 ou les rôles IAM, qui sont partagés avec une entité externe. Cela vous aide à identifier les accès imprévus à vos ressources et données, ce qui constitue un risque de sécurité. Pour en savoir plus, consultez Qu'est-ce qu'IAM Access Analyzer ?.

- Amazon CloudWatch surveille vos AWS ressources et les applications que vous utilisez AWS en temps réel. Vous pouvez collecter et suivre les métriques, créer des tableaux de bord personnalisés, et définir des alarmes qui vous informent ou prennent des mesures lorsqu'une métrique spécifique atteint un seuil que vous spécifiez. Par exemple, vous pouvez CloudWatch suivre l'utilisation du processeur ou d'autres indicateurs de vos EC2 instances Amazon et lancer automatiquement de nouvelles instances en cas de besoin. Pour plus d'informations, consultez le guide de CloudWatch l'utilisateur Amazon.
- Amazon CloudWatch Logs vous permet de surveiller, de stocker et d'accéder à vos fichiers journaux à partir d' EC2 instances Amazon et d'autres sources. CloudTrail CloudWatch Les journaux peuvent surveiller les informations contenues dans les fichiers journaux et vous avertir lorsque certains seuils sont atteints. Vous pouvez également archiver vos données de journaux dans une solution de stockage hautement durable. Pour plus d'informations, consultez le guide de l'utilisateur d'Amazon CloudWatch Logs.

Pour consulter des ressources supplémentaires et les bonnes pratiques en matière de sécurité pour IAM, reportez-vous à la section Bonnes pratiques de sécurité et cas d'utilisation dans Gestion des identités et des accès AWS.

Rubriques

- Journalisation IAM et AWS STS Appels d'API avec AWS CloudTrail
- Suivez les tâches privilégiées dans AWS CloudTrail

Journalisation IAM et AWS STS Appels d'API avec AWS CloudTrail

IAM et AWS STS sont intégrés à AWS CloudTrail un service qui fournit un enregistrement des actions entreprises par un utilisateur ou un rôle IAM. CloudTrail capture tous les appels d'API pour IAM et AWS STS sous forme d'événements, y compris les appels depuis la console et les appels d'API. Si vous créez un historique, vous pouvez activer la diffusion continue des CloudTrail événements vers un compartiment Amazon S3. Si vous ne configurez pas de journal d'activité, vous pouvez toujours afficher les événements les plus récents dans la console CloudTrail dans Historique des événements. Vous pouvez utiliser CloudTrail pour obtenir des informations sur la demande qui a été faite à IAM ou AWS STS. Par exemple, vous pouvez afficher l'adresse IP à partir de laquelle la demande a été effectuée, qui a effectué la demande et quand, ainsi que des détails supplémentaires.

Pour en savoir plus CloudTrail, consultez le Guide de AWS CloudTrail l'utilisateur.

Rubriques

- IAM et AWS STS informations dans CloudTrail
- Journalisation IAM et AWS STS Demandes d'API
- Journalisation des demandes d'API à d'autres AWS services
- Journalisation des événements de connexion utilisateur
- Journalisation des événements de connexion pour les informations d'identification temporaires
- Exemples d'événements d'API IAM dans le journal CloudTrail
- Exemple AWS STS Événements de l'API dans le CloudTrail journal
- Exemples d'événements de connexion dans le journal CloudTrail
- Comportement de la politique d’approbation de rôle IAM

IAM et AWS STS informations dans CloudTrail

CloudTrail est activé Compte AWS lors de la création de votre compte. Lorsqu'une activité se produit dans IAM ou AWS STS, cette activité est enregistrée dans un CloudTrail événement avec d'autres événements de AWS service dans l'historique des événements. Vous pouvez consulter, rechercher et télécharger les événements récents dans votre Compte AWS. Pour plus d'informations, consultez la section Affichage des événements avec l'historique des CloudTrail événements.

Pour un enregistrement continu des événements qui vous concernent Compte AWS, y compris les événements pour IAM AWS STS, et créez une trace. Un journal permet CloudTrail de transmettre des fichiers journaux à un compartiment Amazon S3. Par défaut, lorsque vous créez un journal de suivi dans la console, il s’applique à toutes les régions . Le journal enregistre les événements de toutes les régions de la AWS partition et transmet les fichiers journaux au compartiment Amazon S3 que vous spécifiez. En outre, vous pouvez configurer d'autres AWS services pour analyser plus en profondeur les données d'événements collectées dans les CloudTrail journaux et y donner suite. Pour en savoir plus, consultez :

- Présentation de la création d’un journal d’activité
- CloudTrail Services et intégrations pris en charge
- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers CloudTrail journaux provenant de plusieurs régions et réception de fichiers CloudTrail journaux provenant de plusieurs comptes

Toutes les IAM et les AWS STS actions sont enregistrées CloudTrail et documentées dans les références d'API IAM et les références d'AWS Security Token Service API.

Journalisation IAM et AWS STS Demandes d'API

CloudTrail enregistre toutes les demandes d'API authentifiées dans les opérations IAM et AWS STS API. CloudTrail enregistre également les demandes non authentifiées relatives aux AWS STS actions`AssumeRoleWithWebIdentity`, `AssumeRoleWithSAML` et enregistre les informations fournies par le fournisseur d'identité. Cependant, certaines AWS STS demandes non authentifiées peuvent ne pas être enregistrées car elles ne répondent pas aux attentes minimales d'être suffisamment valides pour être considérées comme des demandes légitimes. CloudTrail n'enregistre pas les AWS STS demandes refusées dans le compte cible pour les hypothèses de rôles entre comptes et les sessions privilégiées initiées via l'`AssumeRoot`API.

Vous pouvez utiliser ces informations journalisées pour mapper les appels effectués par un principal fédéré OIDC ou SAML avec un rôle présumé à l'appelant fédéré externe d'origine. Dans le cas de`AssumeRole`, vous pouvez mapper les appels vers le AWS service d'origine ou vers le compte de l'utilisateur d'origine. La `userIdentity` section des données JSON de l'entrée de CloudTrail journal contient les informations dont vous avez besoin pour mapper la AssumeRole* demande avec un principal de session spécifique. Pour plus d'informations, consultez la section CloudTrail UserIdentity Element dans le Guide de AWS CloudTrail l'utilisateur.

AWS CloudTrail les journaux contiendront des informations MFA lorsque l'utilisateur IAM se connectera avec MFA. Si l'utilisateur IAM assume un rôle IAM, il CloudTrail enregistre également les `sessionContext` attributs des actions effectuées `mfaAuthenticated: true` à l'aide du rôle assumé. Cependant, la CloudTrail journalisation est distincte de ce qu'IAM exige lorsque les appels d'API sont effectués avec les informations d'identification du rôle assumé. Pour en savoir plus, consultez Élément CloudTrail userIdentity.

Par exemple, les appels à l'IAM`CreateUser`, `DeleteRoleListGroups`, et les autres opérations d'API sont tous enregistrés par CloudTrail.

Des exemples de ce type d'entrée de journal sont présentés ultérieurement dans cette rubrique.

Journalisation des demandes d'API à d'autres AWS services

Les demandes authentifiées adressées à d'autres opérations d'API de AWS service sont enregistrées par CloudTrail, et ces entrées de journal contiennent des informations sur l'auteur de la demande.

Par exemple, supposons que vous ayez effectué une demande de liste des instances Amazon EC2 ou de création d'un groupe de déploiement AWS CodeDeploy . Les détails sur la personne ou le service qui a émis la demande sont enregistrés dans l'entrée de journal de cette demande. Ces informations vous permettent de déterminer si la demande a été faite par un utilisateur IAM, un rôle ou un autre AWS service. Utilisateur racine d'un compte AWS

Pour plus de détails sur les informations d'identité utilisateur contenues dans les entrées de CloudTrail journal, consultez la section UserIdentity Element dans le Guide de AWS CloudTrail l'utilisateur.

Journalisation des événements de connexion utilisateur

CloudTrail enregistre les événements de connexion aux Console de gestion AWS outils de développement locaux tels que AWS CLI les SDK, les forums de AWS discussion et. AWS Marketplace CloudTrailenregistre les tentatives de connexion réussies et infructueuses pour les utilisateurs IAM, les principaux fédérés SAML et OIDC et les principaux utilisateurs fédérés. AWS STS

Pour consulter CloudTrail des exemples d'événements relatifs aux connexions réussies et infructueuses des utilisateurs root, voir Exemples d'enregistrements d'événements pour les utilisateurs root dans le Guide de l'AWS CloudTrail utilisateur.

Pour des raisons de sécurité, AWS n'enregistre pas le texte du nom d'utilisateur IAM saisi lorsque l'échec de connexion est dû à un nom d'utilisateur incorrect. Le texte du nom utilisateur est masqué par la valeur `HIDDEN_DUE_TO_SECURITY_REASONS`. Pour en obtenir un exemple, consultez Exemple d'événement d'échec de connexion provoqué par un nom d'utilisateur incorrect, plus loin dans cette rubrique. Le texte du nom d'utilisateur est obscurci car de tels échecs peuvent être causés par des erreurs d'utilisateur. La consignation de ces erreurs pourrait exposer des informations potentiellement sensibles. Par exemple :

- Vous tapez par erreur votre mot de passe dans le champ de nom d'utilisateur.
- Vous choisissez le lien vers la page de connexion de l'un d'entre eux Compte AWS, puis vous saisissez le numéro de compte d'un autre Compte AWS.
- Vous oubliez sur quel compte vous vous connectez et vous tapez par erreur le nom de compte de votre compte de messagerie personnelle, votre identificateur de connexion bancaire ou un autre ID privé.

Journalisation des événements de connexion pour les informations d'identification temporaires

Lorsqu'un mandataire demande des informations d'identification temporaires, le type de mandataire détermine comment CloudTrail consigne l'événement. Cela peut être compliqué lorsqu'un principal endosse un rôle dans un autre compte. Il existe plusieurs appels d'API pour effectuer des opérations liées aux opérations inter-comptes de rôle. Tout d'abord, le principal appelle une AWS STS API pour récupérer les informations d'identification temporaires. Cette opération est enregistrée dans le compte appelant et dans le compte sur lequel l' AWS STS opération est effectuée. Ensuite, le principal utilise le rôle pour effectuer d'autres appels d'API dans le compte du rôle endossé.

Vous pouvez utiliser la clé de condition `sts:SourceIdentity` dans la politique d'approbation de rôle pour exiger des utilisateurs qu'ils spécifient une identité lorsqu'ils endossent un rôle. Par exemple, vous pouvez exiger que les utilisateurs IAM spécifient leur propre nom d'utilisateur comme identité de source. Cela peut vous aider à déterminer quel utilisateur a effectué une action spécifique dans AWS. Pour de plus amples informations, veuillez consulter `sts:SourceIdentity`. Vous pouvez utiliser `sts:RoleSessionName` pour exiger des utilisateurs qu'ils spécifient un nom de session lorsqu'ils endossent un rôle. Cela peut vous aider à différencier les sessions de rôle pour un rôle utilisé par différents acteurs principaux lorsque vous consultez les AWS CloudTrail journaux.

Le tableau suivant montre comment CloudTrail enregistre les différentes informations d'identité utilisateur pour chacune des AWS STS API qui génèrent des informations d'identification temporaires.

Type de principal	API STS	Identité de l'utilisateur dans le CloudTrail journal du compte de l'appelant	Identité de l'utilisateur dans le CloudTrail journal pour le compte du rôle assumé	Identité de l'utilisateur dans le CloudTrail journal pour les appels d'API suivants du rôle
Utilisateur racine d'un compte AWS informations d'identification	GetSessionToken	Identité de l'utilisateur racine	Le compte du propriétaire de rôle est le même que le compte appelant	Identité de l'utilisateur racine

Type de principal	API STS	Identité de l'utilisateur dans le CloudTrai l journal du compte de l'appelant	Identité de l'utilisateur dans le CloudTrail journal pour le compte du rôle assumé	Identité de l'utilisateur dans le CloudTrail journal pour les appels d'API suivants du rôle
Utilisateur racine d'un compte AWS informati ons d'identif ication	AssumeRoot	Session de l’utilisateur racine	Numéro de compte et ID de principal (s’il s’agit d’un utilisateur)	Session de l’utilisateur racine
Utilisateur IAM	GetSessio nToken	Identité d'utilisa teur IAM	Le compte du propriétaire de rôle est le même que le compte appelant	Identité d'utilisa teur IAM
Utilisateur IAM	GetFedera tionToken	Identité d'utilisa teur IAM	Le compte du propriétaire de rôle est le même que le compte appelant	Identité d'utilisa teur IAM
Utilisateur IAM	AssumeRole	Identité d'utilisa teur IAM	Numéro de compte et identifiant principal (s'il s'agit d'un utilisateur), ou principal AWS de service	Identité de rôle uniquement (aucun utilisate ur)

Type de principal	API STS	Identité de l'utilisateur dans le CloudTrai l journal du compte de l'appelant	Identité de l'utilisateur dans le CloudTrail journal pour le compte du rôle assumé	Identité de l'utilisateur dans le CloudTrail journal pour les appels d'API suivants du rôle
Utilisateur authentifié en externe	AssumeRol eWithSAML	n/a	Identité d'utilisa teur SAML	Identité de rôle uniquement (aucun utilisate ur)
Utilisateur authentifié en externe	AssumeRol eWithWebI dentity	n/a	OIDC/Web identité de l'utilisateur	Identité de rôle uniquement (aucun utilisate ur)

CloudTrail considère une action en lecture seule si elle n'a aucun effet mutant sur une ressource. Lors de l'enregistrement d'un événement en lecture seule, CloudTrail supprime les `responseElements` informations du journal. Lorsque l' CloudTrail enregistrement d'un événement n'est pas en lecture seule, l'intégralité `responseElements` est affichée dans l'entrée du journal. Pour les AWS STS API`AssumeRole`,`AssumeRoleWithSAML`, et`AssumeRoleWithWebIdentity`, même si elles sont enregistrées en lecture seule, CloudTrail inclura l'intégralité, `responseElements` sauf `secretAccessKey` dans le journal de ces API.

Le tableau suivant montre comment fonctionne le CloudTrail journal `responseElements` et les `readOnly` informations pour chacune des AWS STS API qui génèrent des informations d'identification temporaires.

API STS	Informations sur les éléments de réponse	Read-only
AssumeRole	Inclus	true
AssumeRoleWithSAML	Inclus	true
AssumeRoleWithWebIdentity	Inclus	true

API STS	Informations sur les éléments de réponse	Read-only
AssumeRoot	Inclus	false
GetFederationToken	Inclus	false
GetSessionToken	Inclus	false

Exemples d'événements d'API IAM dans le journal CloudTrail

CloudTrail les fichiers journaux contiennent des événements au format JSON. Un événement d'API représente une demande d'API unique et inclut des informations sur l'action principale demandée, sur tous les paramètres, et sur la date et l'heure de l'action.

Exemple d'événement d'API IAM dans un fichier CloudTrail journal

L'exemple suivant montre une entrée de CloudTrail journal pour une demande faite pour l'`GetUserPolicy`action IAM.

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:Role-Session-Name",
        "arn": "arn:aws:sts::111122223333:assumed-role/Role-Name/Role-Session-Name",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2024-09-09T17:50:16Z",
                "mfaAuthenticated": "false"
            }
        }
```

```
    },
    "eventTime": "2024-09-09T17:51:44Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "GetUserPolicy",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.101",
    "userAgent": "aws-cli/1.16.96 Python/2.7.8 Linux/10 botocore/1.12.86",
    "requestParameters": {
        "userName": "ExampleIAMUserName",
        "policyName": "ExamplePoliccyName"
    },
    "responseElements": null,
    "requestID": "9EXAMPLE-0c68-11e4-a24e-d5e16EXAMPLE",
    "eventID": "cEXAMPLE-127e-4632-980d-505a4EXAMPLE",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "iam.amazonaws.com"
    }
}
```

À partir de ces informations d'événement, vous pouvez déterminer que la demande a été effectuée pour obtenir une politique utilisateur nommée `ReadOnlyAccess-JaneDoe-201407151307` pour l'utilisateur `JaneDoe`, comme indiqué dans l'élément `requestParameters`. Vous pouvez également voir que la demande a été effectuée par une utilisatrice IAM nommée `JaneDoe` le 15 juillet 2014 à 21:40 (UTC). Dans ce cas, la demande provient de Console de gestion AWS, comme vous pouvez le constater à partir de l'`userAgent`élément.

Exemple AWS STS Événements de l'API dans le CloudTrail journal

CloudTrail les fichiers journaux contiennent des événements au format JSON. Un événement d'API représente une demande d'API unique et inclut des informations sur l'action principale demandée, sur tous les paramètres, et sur la date et l'heure de l'action.

Exemple de compte croisé AWS STS Événements d'API dans les fichiers CloudTrail journaux

L'utilisateur IAM nommé `John` dans le compte 777788889999 appelle l' AWS STS AssumeRoleaction pour assumer le rôle dans le compte 111122223333. `EC2-dev` L'administrateur de compte exige des utilisateurs qu'ils définissent une identité source équivalente à leur nom d'utilisateur lorsqu'ils endossent le rôle. L'utilisateur transmet la valeur d'identité source de `John`.


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "arn": "arn:aws:iam::777788889999:user/John",
    "accountId": "777788889999",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "John"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "John-EC2-dev",
    "sourceIdentity": "John",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2023, 4:07:39 PM"
      },
    "assumedRoleUser": {
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:John-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/John-EC2-dev"
    },
    "sourceIdentity": "John",
    "sessionTokenSize": 1428,
    "sessionTokenUtilization": 34
```

```
  },
  "resources": [
    {
      "ARN": "arn:aws:iam::111122223333:role/EC2-dev",
      "accountId": "111122223333",
      "type": "AWS::IAM::Role"
    }
  ],
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

Le deuxième exemple montre l'entrée de CloudTrail journal du compte de rôle assumé (111122223333) pour la même demande.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AWSAccount",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "accountId": "777788889999"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "John-EC2-dev",
    "sourceIdentity": "John",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2014, 4:07:39 PM"
```

```
    },
    "assumedRoleUser": {
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:John-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/John-EC2-dev"
      },
    "sourceIdentity": "John",
    "sessionTokenSize": 1428,
    "sessionTokenUtilization": 34
  },
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE"
}
```

Exemple AWS STS événement d'API de chaînage de rôles dans le fichier CloudTrail journal

L'exemple suivant montre une entrée de CloudTrail journal pour une demande faite par John Doe dans le compte 1111111111. John a précédemment utilisé son utilisateur `John` pour endosser le rôle `JohnRole1`. Pour cette demande, il utilise les informations d'identification de ce rôle pour endosser le rôle `JohnRole2`. Ceci est connu sous le nom de chaînage de rôles. L'identité source qu'il a définie lorsqu'il a endossé le rôle `John1` persiste dans la demande d'endosser le `JohnRole2`. Si John tente de définir une identité source différente lorsqu'il endosse le rôle, la demande est refusée. John transmet deux balises de session dans la demande. Il définit ces deux balises comme transitives. La demande hérite de la balise `Department` comme transitive car John l'a définie comme transitive lorsqu'il a endossé `JohnRole1`. Pour plus d'informations sur l'identité source, consultez Surveiller et contrôler les actions prises avec les rôles endossés. Pour de plus amples informations sur les clés transitives dans les chaînes de rôles, veuillez consulter Chaînage des rôles avec des balises de session.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIN5ATK5U7KEXAMPLE:JohnRole1",
        "arn": "arn:aws:sts::111111111111:assumed-role/John/JohnRole1",
        "accountId": "111111111111",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2019-10-02T21:50:54Z"
```

```
            },
                "sessionIssuer": {
                "type": "Role",
               "principalId": "AROAIN5ATK5U7KEXAMPLE",
                "arn": "arn:aws:iam::111111111111:role/JohnRole1",
                "accountId": "111111111111",
                "userName": "John"
            },
            "sourceIdentity": "John"
        }
    },
    "eventTime": "2019-10-02T22:12:29Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "123.145.67.89",
    "userAgent": "aws-cli/1.16.248 Python/3.4.7
Linux/4.9.184-0.1.ac.235.83.329.metal1.x86_64 botocore/1.12.239",
    "requestParameters": {
        "incomingTransitiveTags": {
            "Department": "Engineering"
        },
        "tags": [
            {
                "value": "johndoe@example.com",
                "key": "Email"
            },
            {
                "value": "12345",
                "key": "CostCenter"
            }
        ],
        "roleArn": "arn:aws:iam::111111111111:role/JohnRole2",
        "roleSessionName": "Role2WithTags",
        "sourceIdentity": "John",
        "transitiveTagKeys": [
            "Email",
            "CostCenter"
        ],
        "durationSeconds": 3600
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
```

```
            "expiration": "Oct 2, 2019, 11:12:29 PM",
            "sessionToken": "AgoJb3JpZ2luX2VjEB4aCXVzLXdlc3QtMSJHMEXAMPLETOKEN
+//rJb8Lo30mFc5MlhFCEbubZvEj0wHB/mDMwIgSEe9gk/Zjr09tZV7F1HDTMhmEXAMPLETOKEN/iEJ/
rkqngII9//////////
ARABGgw0MjgzMDc4NjM5NjYiDLZjZFKwP4qxQG5sFCryASO4UPz5qE97wPPH1eLMvs7CgSDBSWfonmRTCfokm2FN1+hWUdQ
+C+WKFZb701eiv9J5La2EXAMPLETOKEN/c7S5Iro1WUJ0q3Cxuo/8HUoSxVhQHM7zF7mWWLhXLEQ52ivL
+F6q5dpXu4aTFedpMfnJa8JtkWwG9x1Axj0Ypy2ok8v5unpQGWych1vwdvj6ez1Dm8Xg1+qIzXILiEXAMPLETOKEN/
vQGqu8H+nxp3kabcrtOvTFTvxX6vsc8OGwUfHhzAfYGEXAMPLETOKEN/
L6v1yMM3B1OwFOrQBno1HEjf1oNI8RnQiMNFdUOtwYj7HUZIOCZmjfN8PPHq77N7GJl9lzvIZKQA0Owcjg
+mc78zHCj8y0siY8C96paEXAMPLETOKEN/
E3cpksxWdgs91HRzJWScjN2+r2LTGjYhyPqcmFzzo2mCE7mBNEXAMPLETOKEN/oJy
+2o83YNW5tOiDmczgDzJZ4UKR84yGYOMfSnF4XcEJrDgAJ3OJFwmTcTQICAlSwLEXAMPLETOKEN"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROAIFR7WHDTSOYQYHFUE:Role2WithTags",
            "arn": "arn:aws:sts::111111111111:assumed-role/test-role/Role2WithTags"
        },
        "sourceIdentity": "John",
        "sessionTokenSize": 1652,
        "sessionTokenUtilization": 40
    },
    "requestID": "b96b0e4e-e561-11e9-8b3f-7b396EXAMPLE",
    "eventID": "1917948f-3042-46ec-98e2-62865EXAMPLE",
    "resources": [
        {
            "ARN": "arn:aws:iam::111111111111:role/JohnRole2",
            "accountId": "111111111111",
            "type": "AWS::IAM::Role"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111111111111"
}
```


Exemple AWS web AWS STS Événement d'API dans le fichier CloudTrail journal

L'exemple suivant montre une entrée de CloudTrail journal pour une demande faite par un AWS service appelant une autre API de service à l'aide des autorisations d'un rôle de service. Il affiche l'entrée du CloudTrail journal pour la demande faite dans le compte 777788889999.


```
{
  "eventVersion": "1.04",
  "userIdentity": {
```

```
    "type": "AssumedRole",
    "principalId": "AROAQRSTUVWXYZEXAMPLE:devdsk",
    "arn": "arn:aws:sts::777788889999:assumed-role/AssumeNothing/devdsk",
    "accountId": "777788889999",
    "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2016-11-14T17:25:26Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAQRSTUVWXYZEXAMPLE",
        "arn": "arn:aws:iam::777788889999:role/AssumeNothing",
        "accountId": "777788889999",
        "userName": "AssumeNothing"
      }
    }
  },
  "eventTime": "2016-11-14T17:25:45Z",
  "eventSource": "s3.amazonaws.com",
  "eventName": "DeleteBucket",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.1",
  "userAgent": "[aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67]",
  "requestParameters": {
    "bucketName": "amzn-s3-demo-bucket"
  },
  "responseElements": null,
  "requestID": "EXAMPLE463D56D4C",
  "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "777788889999"
}
```


Exemple SAML AWS STS Événement d'API dans le fichier CloudTrail journal

L'exemple suivant montre une entrée de CloudTrail journal pour une demande faite pour l' AWS STS AssumeRoleWithSAMLaction. La demande inclut les attributs SAML `CostCenter` et `Project` qui sont transmis par l'assertion SAML en tant que balises de session. Ces balises sont définies comme transitives afin qu'elles persistent dans les scénarios de chaînage des rôles. La demande inclut le paramètre d'API facultatif`DurationSeconds`, représenté comme `durationSeconds` dans

le CloudTrail journal, et est définie en `1800` secondes. La demande inclut également l'attribut SAML `sourceIdentity`, qui est transmise dans l'assertion SAML. Si quelqu'un utilise les informations d'identification de session de rôle résultantes pour endosser un autre rôle, cette identité source persiste.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "SAMLUser",
        "principalId": "SampleUkh1i4+ExamplexL/jEvs=:SamlExample",
        "userName": "SamlExample",
        "identityProvider": "bdGOnTesti4+ExamplexL/jEvs="
    },
    "eventTime": "2023-08-28T18:30:58Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoleWithSAML",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "aws-internal/3 aws-sdk-java/1.12.479
 Linux/5.10.186-157.751.amzn2int.x86_64 OpenJDK_64-Bit_Server_VM/17.0.7+11 java/17.0.7
 kotlin/1.3.72 vendor/Amazon.com_Inc. cfg/retry-mode/standard",
    "requestParameters": {
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyAssignedRoleSessionName",
        "sourceIdentity": "MySAMLUser",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn",
            "Department": "Engineering"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "roleArn": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::444455556666:saml-provider/Shibboleth",
        "durationSeconds": 1800
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Aug 28, 2023, 7:00:58 PM"
```

```
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROAD35QRSTUVWEXAMPLE:MyAssignedRoleSessionName",
            "arn": "arn:aws:sts::444455556666:assumed-role/SAMLTestRoleShibboleth/
MyAssignedRoleSessionName"
        },
        "packedPolicySize": 34,
        "sessionTokenSize": 1428,
        "sessionTokenUtilization": 34,
        "subject": "SamlExample",
        "subjectType": "transient",
        "issuer": "https://server.example.com/idp/shibboleth",
        "audience": "https://signin.aws.amazon.com/saml",
        "nameQualifier": "bdGOnTesti4+ExamplexL/jEvs=",
        "sourceIdentity": "MySAMLUser"
    },
    "requestID": "6EXAMPLE-e595-11e5-b2c7-c974fEXAMPLE",
    "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth"
        },
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::SAMLProvider",
            "ARN": "arn:aws:iam::444455556666:saml-provider/test-saml-provider"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "444455556666",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "sts.us-east-2.amazonaws.com"
    }
}
```

Exemple OIDC AWS STS Événement d'API dans le fichier CloudTrail journal

L'exemple suivant montre une entrée de CloudTrail journal pour une demande faite pour l' AWS STS AssumeRoleWithWebIdentityaction. La requête inclut les attributs `CostCenter` et `Project` qui sont transmis par le jeton du fournisseur d'identité (IdP) OpenID Connect (OIDC) en tant que balises de session. Ces balises sont définies comme transitives afin qu'elles persistent dans le chaînage des rôles. La demande inclut l'attribut `sourceIdentity` du jeton du fournisseur d'identité. Si quelqu'un utilise les informations d'identification de session de rôle résultantes pour endosser un autre rôle, cette identité source persiste.

L'entrée du CloudTrail journal contient également un `additionalEventData` champ avec un `identityProviderConnectionVerificationMethod` attribut. Cet attribut indique la méthode AWS utilisée pour vérifier la connexion avec le fournisseur OIDC. La valeur de l'attribut sera `IAMTrustStore` ou `Thumbprint`. La `IAMTrustStore` valeur indique que la connexion avec l'IdP OIDC a été vérifiée avec AWS succès à l'aide de notre bibliothèque d'autorités de certification (CA) racines fiables. La `Thumbprint` valeur indique que l'empreinte numérique du certificat a été AWS utilisée dans la configuration IdP pour vérifier le certificat du serveur OIDC IdP.

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "WebIdentityUser",
    "principalId": "arn:aws:iam::444455556666:oidc-provider/<issuer url of OIDC
 provider>:<id of application>:<id of user>",
    "userName": "<id of user>",
    "identityProvider": "arn:aws:iam::444455556666:oidc-provider/<issuer url of OIDC
 provider>"
  },
  "eventTime": "2024-07-09T15:41:37Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRoleWithWebIdentity",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/2.13.29 Python/3.11.6 Windows/10 exe/AMD64 prompt/off command/
sts.assume-role-with-web-identity",
  "requestParameters": {
    "roleArn": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
    "roleSessionName": "<assigned role session name>",
    "sourceIdentity": "MyWebIdentityUser",
    "durationSeconds": 3600,
    "principalTags": {
```

```
      "CostCenter": "24680",
      "Project": "Pegasus"
    },
    "transitiveTagKeys": [
      "CostCenter",
      "Project"
    ]
  },
  "responseElements": {
    "credentials": {
      "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
      "sessionToken": "<encoded session token blob>",
      "expiration": "Jul 9, 2024, 4:41:37 PM"
    },
    "subjectFromWebIdentityToken": "<id of user>",
    "sourceIdentity": "MyWebIdentityUser",
    "assumedRoleUser": {
      "assumedRoleId": "AROA123456789EXAMPLE:<assigned role session name>",
      "arn": "arn:aws:sts::444455556666:assumed-role/FederatedWebIdentityRole/<assigned
role session name>"
    },
    "provider": "arn:aws:iam::444455556666:oidc-provider/<issuer url of OIDC
provider>",
    "audience": "<id of application>",
    "sessionTokenSize": 1536,
    "sessionTokenUtilization": 37
  },
  "additionalEventData": {
    "identityProviderConnectionVerificationMethod": "IAMTrustStore"
  },
  "requestID": "aEXAMPLE-0b26-40df-8973-c7012EXAMPLE",
  "eventID": "aEXAMPLE-ee29-4ac0-a0ed-3f5c5EXAMPLE",
  "readOnly": true,
  "resources": [
    {
      "accountId": "444455556666",
      "type": "AWS::IAM::Role",
      "ARN": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "444455556666",
  "eventCategory": "Management",
```

```
  "tlsDetails": {
    "tlsVersion": "TLSv1.3",
    "cipherSuite": "TLS_AES_128_GCM_SHA256",
    "clientProvidedHostHeader": "sts.us-east-2.amazonaws.com"
  }
}
```

Exemple AWS STS Événement d'API utilisant le point de terminaison global dans le fichier CloudTrail journal

Pour les requêtes adressées au AWS Security Token Service point de terminaison global (`https://sts.amazonaws.com`), AWS STS inclut des champs de AWS CloudTrail journal supplémentaires : `endpointType` et`awsServingRegion`.AWS STS Ces champs apparaissent sous l'`additionalEventDataRequestDetails`élément pour enregistrer le type de serveur Région AWS et de point de terminaison appelé. Le champ `endpointType` peut avoir une valeur de `global` ou `regional` pour indiquer le type de point de terminaison global qui a répondu à la demande. Pour plus d'informations sur les modifications AWS STS globales des points de terminaison, consultezAWS STS Régions et points de terminaison.

> **Note**
>
> AWS CloudTrail les journaux des demandes adressées au terminal AWS STS mondial seront envoyés à la région de l'Est des États-Unis (Virginie du Nord). CloudTrail les journaux des demandes traitées par les terminaux AWS STS régionaux continueront d'être enregistrés dans CloudTrail leur région respective.

L'exemple suivant montre une entrée de CloudTrail journal pour une AWS STS demande adressée au point de terminaison global (`https://sts.amazonaws.com`) qui provient de la région Europe (Stockholm) - eu-north-1. La valeur du `endpointType` champ `global` indique que la AWS STS demande a été traitée par le point de terminaison mondial de la région Europe (Stockholm).

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA123456789EXAMPLE:developer",
        "arn": "arn:aws:sts::777788889999:assumed-role/Admin/developer",
        "accountId": "777788889999",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
```

```
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA123456789EXAMPLE",
                "arn": "arn:aws:iam::777788889999:role/Admin",
                "accountId": "777788889999",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2025-02-12T21:44:28Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2025-02-12T22:16:48Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.15.33 Python/3.11.8 Linux/5.10.233-204.894.amzn2int.x86_64
 exe/x86_64.amzn.2 prompt/off command/sts.assume-role",
    "requestParameters": {
        "roleArn": "arn:aws:iam::777788889999:role/test-role",
        "roleSessionName": "test-global-assume-role"
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Feb 12, 2025, 11:16:48 PM"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROA987654321EXAMPLE:test-global-assume-role",
            "arn": "arn:aws:sts::777788889999:assumed-role/test-role/test-global-
assume-role"
        },
        "sessionTokenSize": 1096,
        "sessionTokenUtilization": 26
    },
    "additionalEventData": {
        "RequestDetails": {
            "awsServingRegion": "eu-north-1",
            "endpointType": "global"
```

```
        }
    },
    "requestID": "EXAMPLE7-2497-457a-9586-f21feEXAMPLE",
    "eventID": "EXAMPLEc-3d26-4c3a-9c94-722a9EXAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "777788889999",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::777788889999:role/test-role"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "777788889999",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "sts-global.eu-north-1.amazonaws.com"
    }
}
```

À titre de comparaison, l'exemple suivant montre une entrée de CloudTrail journal pour une AWS STS demande adressée au point de terminaison régional (`https://sts.us-west-2.amazonaws.com`) qui était desservi par le point de terminaison régional de la région Europe (Stockholm) - eu-north-1. La valeur du `endpointType` champ `regional` indique que la AWS STS demande a été traitée par le point de terminaison mondial de la région Europe (Stockholm).

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA123456789EXAMPLE:developer",
        "arn": "arn:aws:sts::777788889999:assumed-role/Admin/developer",
        "accountId": "777788889999",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA123456789EXAMPLE",
                "arn": "arn:aws:iam::777788889999:role/Admin",
                "accountId": "777788889999",
```

```
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2025-02-12T21:44:28Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2025-02-12T22:16:30Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "eu-north-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.15.33 Python/3.11.8 Linux/5.10.233-204.894.amzn2int.x86_64
 exe/x86_64.amzn.2 prompt/off command/sts.assume-role",
    "requestParameters": {
        "roleArn": "arn:aws:iam::777788889999:role/test-role",
        "roleSessionName": "test-global-assume-role"
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Feb 12, 2025, 11:16:30 PM"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROA987654321EXAMPLE:test-global-assume-role",
            "arn": "arn:aws:sts::777788889999:assumed-role/test-role/test-global-
assume-role"
        },
        "sessionTokenSize": 1096,
        "sessionTokenUtilization": 26
    },
    "additionalEventData": {
        "RequestDetails": {
            "endpointType": "regional",
            "awsServingRegion": "eu-north-1"
        }
    },
    "requestID": "EXAMPLEd-2116-4cd7-bd72-9f72fEXAMPLE",
    "eventID": "EXAMPLEd-219a-48ed-bc54-00e3cEXAMPLE",
    "readOnly": true,
    "resources": [
```

```
        {
            "accountId": "777788889999",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::777788889999:role/test-role"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "777788889999",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "sts.eu-north-1.amazonaws.com"
    }
}
```

Exemples d'événements de connexion dans le journal CloudTrail

CloudTrail les fichiers journaux contiennent des événements au format JSON. Un événement de connexion représente une seule demande de connexion et inclut des informations sur le principal de connexion, la région, ainsi que la date et l'heure de l'action.

Exemple d'événement de connexion réussie dans le fichier CloudTrail journal

L'exemple suivant montre une entrée de CloudTrail journal pour un événement de connexion réussi.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/John",
    "accountId": "111122223333",
    "userName": "John"
  },
  "eventTime": "2014-07-16T15:49:27Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.110",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
```

```
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Success"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/s3/  ",
    "MFAUsed": "No"
  },
  "eventID": "3fcfb182-98f8-4744-bd45-10a395ab61cb"
}
```

L'exemple suivant montre une entrée de CloudTrail journal pour une demande de code d'autorisation réussie.

```
{
"eventVersion": "1.11",
"userIdentity": {
"type": "AssumedRole",
"principalId": "AROATJHQDX737YZP72NTF:thesjain-Isengard",
"arn": "arn:aws:sts::225989345271:assumed-role/Admin/thesjain-Isengard",
"accountId": "225989345271",
"sessionContext": {
"sessionIssuer": {
"type": "Role",
"principalId": "AROATJHQDX737YZP72NTF",
"arn": "arn:aws:iam::225989345271:role/Admin",
"accountId": "225989345271",
"userName": "Admin"
},
"attributes": {
"creationDate": "2025-11-17T22:50:14Z",
"mfaAuthenticated": "false"
}
}
},
"eventTime": "2025-11-17T22:51:32Z",
"eventSource": "signin.amazonaws.com",
"eventName": "AuthorizeOAuth2Access",
"awsRegion": "us-east-1",
"sourceIPAddress": "52.94.133.136",
```

```
"userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/142.0.0.0 Safari/537.36",
"requestParameters": {
"scope": "openid",
"redirect_uri": "http://127.0.0.1:53037/oauth/callback",
"code_challenge_method": "SHA-256",
"client_id": "arn:aws:signin:::devtools/same-device"
},
"responseElements": null,
"additionalEventData": {
"success": "true",
"x-amzn-vpce-id": ""
},
"requestID": "e2854c76-1cba-4360-9fd1-5037b591466b",
"eventID": "59e1720d-3deb-44ff-933d-6828be2a860a",
"readOnly": true,
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "225989345271",
"eventCategory": "Management",
"tlsDetails": {
"tlsVersion": "TLSv1.3",
"cipherSuite": "TLS_AES_128_GCM_SHA256",
"clientProvidedHostHeader": "us-east-1.signin.aws.amazon.com"
}
}
```


L'exemple suivant montre une entrée de CloudTrail journal pour une demande de création de jeton OAuth2 réussie.


```
{
"eventVersion": "1.11",
"userIdentity": {
"type": "AssumedRole",
"principalId": "AROATJHQDX737YZP72NTF:jacobjoj-Isengard",
"arn": "arn:aws:sts::225989345271:assumed-role/Admin/jacobjoj-Isengard",
"accountId": "225989345271",
"sessionContext": {
"sessionIssuer": {
"type": "Role",
```

```
"principalId": "AROATJHQDX737YZP72NTF",
"arn": "arn:aws:iam::225989345271:role/Admin",
"accountId": "225989345271",
"userName": "Admin"
},
"attributes": {
"creationDate": "2025-11-18T20:38:10Z",
"mfaAuthenticated": "false"
}
}
},
"eventTime": "2025-11-18T20:38:44Z",
"eventSource": "signin.amazonaws.com",
"eventName": "CreateOAuth2Token",
"awsRegion": "us-east-1",
"sourceIPAddress": "15.248.6.6",
"userAgent": "aws-cli/2.32.0 md/awscrt#0.28.4 ua/2.1 os/macos#24.6.0 md/arch#arm64
 lang/python#3.13.9 md/pyimpl#CPython m/b,AA,Z,E cfg/retry-mode#standard md/
installer#exe sid/35033f4ca1bd md/prompt#off md/command#login",
"requestParameters": {
"client_id": "arn:aws:signin:::devtools/same-device"
},
"responseElements": null,
"additionalEventData": {
"success": "true",
"x-amzn-vpce-id": ""
},
"requestID": "94562943-c85b-4dc1-bf72-43b0fd42d6de",
"eventID": "0b338fac-6a10-4740-b34d-1bb6923e799e",
"readOnly": true,
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "225989345271",
"eventCategory": "Management",
"tlsDetails": {
"tlsVersion": "TLSv1.3",
"cipherSuite": "TLS_AES_128_GCM_SHA256",
"clientProvidedHostHeader": "us-east-1.signin.aws.amazon.com"
}
}
```

Pour plus de détails sur les informations contenues dans les fichiers CloudTrail journaux, consultez la section Référence des CloudTrail événements dans le Guide de AWS CloudTrail l'utilisateur.

Exemple d'événement d'échec de connexion dans le fichier CloudTrail journal

L'exemple suivant montre une entrée de CloudTrail journal pour un échec de connexion.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/JaneDoe",
    "accountId": "111122223333",
    "userName": "JaneDoe"
  },
  "eventTime": "2014-07-08T17:35:27Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.100",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "Failed authentication",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/sns",
    "MFAUsed": "No"
  },
  "eventID": "11ea990b-4678-4bcd-8fbe-62509088b7cf"
}
```

À partir de ces informations, vous pouvez déterminer que la tentative de connexion a été effectuée par une utilisatrice IAM nommée `JaneDoe`, comme illustré dans l'élément `userIdentity`. Vous pouvez également voir que la tentative de connexion a échoué, comme montré dans l'élément `responseElements`. Vous pouvez voir que `JaneDoe` a essayé de se connecter à la console Amazon SNS à 17:35 (UTC) le 8 juillet 2014.

Exemple d'événement d'échec de connexion provoqué par un nom d'utilisateur incorrect

L'exemple suivant montre une entrée de CloudTrail journal pour un échec de connexion dû à la saisie d'un nom d'utilisateur incorrect par l'utilisateur. AWS masque le `userName` texte `HIDDEN_DUE_TO_SECURITY_REASONS` pour éviter d'exposer des informations potentiellement sensibles.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "accountId": "123456789012",
    "accessKeyId": "",
    "userName": "HIDDEN_DUE_TO_SECURITY_REASONS"
  },
  "eventTime": "2015-03-31T22:20:42Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "No username found in supplied account",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "LoginTo": "https://console.aws.amazon.com/console/home?state=hashArgs
%23&isauthcode=true",
    "MobileVersion": "No",
    "MFAUsed": "No"
  },
  "eventID": "a7654656-0417-45c6-9386-ea8231385051",
  "eventType": "AwsConsoleSignin",
  "recipientAccountId": "123456789012"
}
```

Comportement de la politique d’approbation de rôle IAM

Le 21 septembre 2022, AWS a apporté des modifications au comportement de la politique de confiance des rôles IAM afin d'exiger des autorisations explicites dans une politique de confiance des

rôles lorsqu'un rôle s'assume lui-même. Les rôles IAM de la liste d'autorisation des comportements hérités comportent un EventData champ supplémentaire à des fins explicites TrustGrant pour les `AssumeRole` événements. La valeur de `explicitTrustGrant` est fausse lorsqu'un rôle figurant dans l'ancienne liste d'autorisation suppose qu'il utilise le comportement existant. Lorsqu'un rôle figurant dans l'ancienne liste d'autorisation assume lui-même son rôle, mais que le comportement de la politique d'approbation des rôles a été mis à jour pour autoriser explicitement le rôle à s'assumer lui-même, la valeur de `explicitTrustGrant` est vraie.

Seul un très petit nombre de rôles IAM figurent sur la liste d'autorisation pour le comportement existant, et ce champ n'est présent dans les CloudTrail journaux de ces rôles que lorsqu'ils assument eux-mêmes leur rôle. Dans la plupart des cas, il n'est pas nécessaire qu'un rôle IAM s'assume de lui-même. AWS recommande de mettre à jour vos processus, votre code ou vos configurations pour supprimer ce comportement ou de mettre à jour vos politiques de confiance des rôles pour autoriser explicitement ce comportement. Pour plus d'informations, consultez Annonce d'une mise à jour du comportement de la politique d'approbation dans les rôles IAM.

Suivez les tâches privilégiées dans AWS CloudTrail

Le compte AWS Organizations de gestion ou un compte d'administrateur délégué pour IAM peut effectuer certaines tâches d'utilisateur root sur les comptes membres en utilisant un accès root à court terme. Les sessions privilégiées de courte durée vous fournissent des informations d'identification temporaires que vous pouvez définir pour effectuer des actions privilégiées sur un compte membre de votre organisation. Vous pouvez utiliser les étapes suivantes pour identifier les actions effectuées par le compte de gestion ou un administrateur délégué au cours de la session `sts:AssumeRoot`.

> **Note**
>
> Le point de terminaison global n'est pas pris en charge pour`sts:AssumeRoot`. CloudTrail enregistre les `ConsoleLogin` événements dans la région spécifiée pour le point de terminaison.

Pour suivre les actions effectuées par une session privilégiée dans les CloudTrail journaux

1. Trouvez l'`AssumeRoot`événement dans vos CloudTrail journaux. Cet événement est généré lorsque votre compte de gestion ou l'administrateur délégué d'IAM obtient un ensemble d'informations d'identification à court terme de `sts:AssumeRoot`.

Dans l'exemple suivant, l' CloudTrail événement pour AssumeRoot est enregistré dans le `eventName` champ.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:JohnRole1",
        "arn": "arn:aws:sts::111111111111:assumed-role/John/JohnRole1",
        "accountId": "111111111111",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111111111111:role/John",
                "accountId": "111111111111",
                "userName": "Admin2"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2024-10-25T20:45:28Z",
                "mfaAuthenticated": "false"
            },
            "assumedRoot": "true"
        }
    },
    "eventTime": "2024-10-25T20:52:11Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoot",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "requestParameters": {
        "targetPrincipal": "222222222222",
        "taskPolicyArn": {
            "arn": "arn:aws:iam::aws:policy/root-task/S3UnlockBucketPolicy"
        }
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
```

```
            "expiration": "Oct 25, 2024, 9:07:11 PM"
        }
    }
}
```

Pour savoir comment accéder à vos CloudTrail journaux, consultez la section Obtenir et consulter vos fichiers CloudTrail journaux dans le guide de AWS CloudTrail l'utilisateur.

2. Dans le journal des CloudTrail événements, recherchez celui `targetPrincipal` qui indique les actions effectuées sur le compte du membre et celui `accessKeyId` qui est propre à la `AssumeRoot` session.

   Dans l'exemple suivant, `targetPrincipal` c'est 222222222222 et c'est EXAMPLE. `accessKeyId` ASIAIOSFODNN7

```
"eventTime": "2024-10-25T20:52:11Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoot",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "requestParameters": {
        "targetPrincipal": "222222222222",
        "taskPolicyArn": {
            "arn": "arn:aws:iam::aws:policy/root-task/S3UnlockBucketPolicy"
        }
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
            "expiration": "Oct 25, 2024, 9:07:11 PM"
    }
}
```

3. Dans les CloudTrail journaux du principal cible, recherchez l'ID de clé d'accès correspondant à la `accessKeyId` valeur de l'`AssumeRoot`événement. Utilisez les valeurs des champs `eventName` pour déterminer les tâches privilégiées effectuées pendant la session `AssumeRoot`. Plusieurs tâches privilégiées peuvent être effectuées au cours d'une même session. La durée maximale de la session `AssumeRoot` est de 900 secondes (15 minutes).

   Dans l'exemple suivant, le compte de gestion ou l'administrateur délégué a supprimé la politique basée sur les ressources pour un compartiment Amazon S3.

```
{
    "eventVersion": "1.10",
    "userIdentity": {
        "type": "Root",
        "principalId": "222222222222",
        "arn": "arn:aws:iam::222222222222:root",
        "accountId": "222222222222",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "creationDate": "2024-10-25T20:52:11Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2024-10-25T20:53:47Z",
    "eventSource": "s3.amazonaws.com",
    "eventName": "DeleteBucketPolicy",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "requestParameters": {
        "bucketName": "resource-policy-John",
        "Host": "resource-policy-John.s3.amazonaws.com",
        "policy": ""
    },
    "responseElements": null,
    "requestID": "1234567890abcdef0",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "readOnly": false,
    "resources": [
        {
            "accountId": "222222222222",
            "type": "AWS::S3::Bucket",
            "ARN": "arn:aws:s3:::resource-policy-John"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "222222222222",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
```

```
        "clientProvidedHostHeader": "resource-policy-John.s3.amazonaws.com"
    }
}
```

Validation de conformité pour Gestion des identités et des accès AWS

Third-party les auditeurs évaluent la sécurité et la conformité de Gestion des identités et des accès AWS (IAM) dans le cadre de multiples programmes de AWS conformité. Il s'agit notamment des certifications SOC, PCI, FedRAMP, ISO et autres.

Notre nouvelle expérience AWS d'inscription n'est pas conçue pour les charges de travail réglementées. Si vous utilisez notre nouvelle interface d' AWS inscription, mais que vous souhaitez l'utiliser AWS pour des charges de travail réglementées, vous pouvez vous inscrire à des fonctionnalités AWS (avancées) ou activer des fonctionnalités avancées pour votre AWS environnement.

Pour savoir si un s' Service AWS inscrit dans le cadre de programmes de conformité spécifiques, consultez Services AWS la section Étendue par programme de conformité et choisissez le programme de conformité qui vous intéresse. Pour des informations générales, voir Programmes de AWS conformité Programmes .

Vous pouvez télécharger des rapports d'audit tiers à l'aide de AWS Artifact. Pour plus d'informations, voir Téléchargement de rapports dans AWS Artifact .

Votre responsabilité en matière de conformité lors de l'utilisation Services AWS dépend de la sensibilité de vos données, des objectifs de conformité de votre entreprise et des lois et réglementations applicables. Pour plus d'informations sur votre responsabilité en matière de conformité lors de l'utilisation Services AWS, consultez AWS la documentation de sécurité.

Résilience dans Gestion des identités et des accès AWS

L'infrastructure AWS mondiale est construite autour des AWS régions et des zones de disponibilité. AWS Les régions disposent de plusieurs zones de disponibilité physiquement séparées et isolées, connectées par un réseau à faible latence, à haut débit et hautement redondant. Pour plus d'informations sur AWS les régions et les zones de disponibilité, consultez la section Infrastructure AWS mondiale.

Gestion des identités et des accès AWS (IAM) et AWS Security Token Service (AWS STS) sont des services régionaux autonomes disponibles dans le monde entier.

L'IAM est un élément essentiel Service AWS. Chaque opération effectuée AWS doit être authentifiée et autorisée par IAM. IAM vérifie chaque demande en fonction des identités et des politiques stockées dans IAM pour déterminer si la demande est autorisée ou refusée. IAM a été conçu avec un plan de contrôle et un plan de données séparés afin que le service s'authentifie même en cas de défaillance inattendue. Les ressources IAM utilisées dans les autorisations, telles que les rôles et les politiques, sont stockées dans le plan de contrôle. Les clients IAM peuvent modifier la configuration de ces ressources en utilisant des opérations IAM telles que `DeletePolicy` et `AttachRolePolicy`. Ces demandes de modification de configuration sont envoyées au plan de contrôle. Il existe un seul plan de contrôle IAM pour tous les avions commerciaux Régions AWS, situé dans la région de l'est des États-Unis (Virginie du Nord). Le système IAM propage ensuite les modifications de configuration aux plans de données IAM dans chaque Région AWS activée. Le plan de données IAM est essentiellement une réplique en lecture seule des données de configuration du plan de contrôle IAM. Chacun Région AWS dispose d'une instance totalement indépendante du plan de données IAM, qui effectue l'authentification et l'autorisation pour les demandes provenant de la même région. Dans chaque région, le plan de données IAM est réparti sur au moins trois zones de disponibilité et possède une capacité suffisante pour tolérer la perte d'une zone de disponibilité sans nuire au client. Les plans de contrôle et de données IAM ont été conçus pour un temps d'interruption planifié nul, toutes les mises à jour logicielles et les opérations de mise à l'échelle étant effectuées de manière invisible pour les clients.

Dans les régions activées par défaut, les demandes adressées au point de terminaison AWS STS global sont automatiquement traitées dans la même région d'où provient la demande. Dans les régions optionnelles, les demandes adressées au point de terminaison AWS STS mondial sont traitées par une seule entité Région AWS, celle de l'est des États-Unis (Virginie du Nord). Vous pouvez utiliser un point de AWS STS terminaison régional pour réduire la latence ou fournir une redondance supplémentaire à vos applications. Pour en savoir plus, veuillez consulter la section Gérer AWS STS dans un Région AWS.

Certains événements peuvent interrompre la communication Régions AWS entre les réseaux. Cependant, même lorsque vous ne pouvez pas communiquer avec le point de terminaison IAM global, vous AWS STS pouvez toujours authentifier les principaux IAM et IAM peut autoriser vos demandes. Les détails spécifiques d'un événement qui interrompt la communication détermineront votre capacité à accéder aux AWS services. Dans la plupart des cas, vous pouvez continuer à utiliser les informations d'identification IAM dans votre AWS environnement. Les conditions suivantes peuvent s'appliquer à un événement qui interrompt la communication.

Clés d'accès pour utilisateurs IAM

Vous pouvez vous authentifier indéfiniment dans une région avec les clés d'accès longue durée pour utilisateurs IAM. Lorsque vous utilisez le AWS Command Line Interface et APIs, vous pouvez fournir des clés AWS d'accès afin AWS de vérifier votre identité dans les demandes programmatiques.

> ⚠ Important
>
> Selon les bonnes pratiques, nous recommandons à vos utilisateurs de se connecter avec des informations d'identification temporaires plutôt qu'avec des clés d'accès longue durée.

Informations d'identification temporaires

Vous pouvez demander de nouvelles informations d'identification temporaires auprès du point de terminaison du service AWS STS régional pendant au moins 24 heures. Les opérations d'API suivantes génèrent des informations d'identification temporaires.

- AssumeRole
- AssumeRoleWithWebIdentity
- AssumeRoleWithSAML
- GetFederationToken
- GetSessionToken

Principaux et autorisations

- Il se peut que vous ne puissiez pas ajouter, modifier ou supprimer des principaux ou des autorisations dans IAM.
- Il se peut que vos informations d'identification ne reflètent pas les modifications que vous avez récemment apportées à vos autorisations dans IAM. Pour de plus amples informations, veuillez consulter Les modifications que j'apporte ne sont pas toujours visibles immédiatement.

Console de gestion AWS

- Vous pouvez utiliser un point de terminaison de connexion régional pour vous connecter à l' Console de gestion AWS en tant qu'utilisateur IAM. Les points de terminaison de connexion régionaux ont le format d'URL suivant.

  `https://{Account ID}.signin.aws.amazon.com/console?region={Region}`

  Exemple : https://111122223333.signin.aws.amazon.com /console ? région=us-ouest-2

- Il se peut que vous ne puissiez pas terminer l'authentification multifactorielle (MFA) Universal 2nd Factor (U2F).

Bonnes pratiques pour la résilience IAM

AWS a intégré la résilience dans Régions AWS les zones de disponibilité. Lorsque vous respectez les bonnes pratiques IAM suivantes dans les systèmes qui interagissent avec votre environnement, vous profitez de cette résilience.

1. Utilisez un point de terminaison de service AWS STS régional au lieu du point de terminaison global par défaut.
2. Passez en revue la configuration de votre environnement pour les ressources vitales qui créent ou modifient régulièrement des ressources IAM, et préparez une solution de secours qui utilise les ressources IAM existantes.

Sécurité de l'infrastructure dans Gestion des identités et des accès AWS

En tant que service géré, AWS Identity and Access Management est protégé par la sécurité du réseau AWS mondial. Pour plus d'informations sur les services AWS de sécurité et sur la manière dont AWS l'infrastructure est protégée, consultez la section Sécurité du AWS cloud. Pour concevoir votre AWS environnement en utilisant les meilleures pratiques en matière de sécurité de l'infrastructure, consultez la section Protection de l'infrastructure dans le cadre AWS bien architecturé du pilier de sécurité.

Vous utilisez des appels d'API AWS publiés pour accéder à IAM via le réseau. Les clients doivent prendre en charge les éléments suivants :

- Protocole TLS (Transport Layer Security). Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Ses suites de chiffrement PFS (Perfect Forward Secrecy) comme DHE (Ephemeral Diffie-Hellman) ou ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La plupart des systèmes modernes tels que Java 7 et les versions ultérieures prennent en charge ces modes.

IAM est accessible par programmation à l'aide de l'API HTTPS IAM, qui vous permet d'effectuer des demandes HTTPS directement au service. L'API Query renvoie des informations sensibles, y compris des informations d'identification de sécurité. Par conséquent, vous devez utiliser HTTPS avec toutes

les demandes d'API. Lorsque vous utilisez l'API HTTPS, vous devez inclure du code pour signer numériquement les demandes à l'aide de vos informations d'identification.

Vous pouvez appeler ces opérations d'API à partir de n'importe quel emplacement sur le réseau, mais IAM prend en charge les politiques d'accès basées sur les ressources, ce qui peut inclure des restrictions en fonction de l'adresse IP source. Vous pouvez également utiliser les politiques IAM pour contrôler l'accès depuis des points de terminaison Amazon Virtual Private Cloud (Amazon VPC) spécifiques ou spécifiques. VPCs En fait, cela isole l'accès réseau à une ressource IAM donnée uniquement du VPC spécifique au sein du réseau. AWS

Analyse de la configuration et des vulnérabilités dans Gestion des identités et des accès AWS

AWS gère les tâches de sécurité de base telles que l'application de correctifs au système d'exploitation client et aux bases de données, la configuration du pare-feu et la reprise après sinistre. Ces procédures ont été vérifiées et certifiées par les tiers appropriés. Pour plus de détails, consultez les ressources suivantes :

- Modèle de responsabilité partagée
- Amazon Web Services : Présentation des procédures de sécurité (livre blanc)

Les ressources suivantes traitent également de la configuration et de l'analyse des vulnérabilités dans Gestion des identités et des accès AWS (IAM) :

- Validation de conformité pour Gestion des identités et des accès AWS
- Bonnes pratiques de sécurité et cas d'utilisation dans Gestion des identités et des accès AWS

AWS politiques gérées pour Gestion des identités et des accès AWS, IAM Access Analyzer et gestionnaire d'accès aux comptes

Une politique AWS gérée est une politique autonome qui est créée et administrée par AWS. AWS les politiques gérées sont conçues pour fournir des autorisations pour de nombreux cas d'utilisation courants afin que vous puissiez commencer à attribuer des autorisations aux utilisateurs, aux groupes et aux rôles.

N'oubliez pas que les politiques AWS gérées peuvent ne pas accorder les autorisations de moindre privilège pour vos cas d'utilisation spécifiques, car elles sont disponibles pour tous les AWS clients. Nous vous recommandons de réduire encore les autorisations en définissant des politiques gérées par le client qui sont propres à vos cas d'utilisation.

Vous ne pouvez pas modifier les autorisations définies dans les politiques AWS gérées. Si les autorisations définies dans une politique AWS gérée sont AWS mises à jour, la mise à jour affecte toutes les identités principales (utilisateurs, groupes et rôles) auxquelles la politique est attachée. AWS est le plus susceptible de mettre à jour une politique AWS gérée lorsqu'une nouvelle politique Service AWS est lancée ou que de nouvelles opérations d'API deviennent disponibles pour des services existants.

Pour plus d'informations, consultez Politiques gérées par AWS dans le Guide de l'utilisateur IAM.

IAMReadOnlyAccess

Utilisez la politique gérée `IAMReadOnlyAccess` pour autoriser l'accès en lecture seule aux ressources IAM. Cette politique accorde l'autorisation d'obtenir et de répertorier toutes les ressources IAM. Elle permet d'afficher les détails et les rapports d'activité pour les utilisateurs, les groupes, les rôles, les politiques, les fournisseurs d'identité et les dispositifs MFA. Elle n'inclut pas la possibilité de créer ou de supprimer des ressources ou d'accéder aux ressources IAM Access Analyzer. Consultez la politique pour obtenir la liste complète des services et des actions que prend en charge cette politique.

IAMUserChangePassword

Utilisez la politique gérée par l'interface `IAMUserChangePassword` pour autoriser les utilisateurs IAM à modifier leur mot de passe.

Vous configurez vos Account settings (paramètres de compte) IAM et la Password policy (politique de mot de passe) pour autoriser les utilisateurs IAM à modifier le mot de passe de leur compte IAM. Lorsque vous autorisez cette action, IAM attache la politique à chaque utilisateur.

Pour consulter la politique JSON, consultez IAMUserChangePassword le AWS Managed Policy Reference Guide.

IAMAccessAnalyzerFullAccess

Utilisez la politique `IAMAccessAnalyzerFullAccess` AWS gérée pour autoriser vos administrateurs à accéder à IAM Access Analyzer.

Groupes d'autorisations

Cette politique est groupée en instructions basées sur le jeu d'autorisations fourni.

- IAM Access Analyzer : octroie des autorisations administratives totales à l'ensemble des ressources dans IAM Access Analyzer.
- Créer un rôle lié au service : autorise l'administrateur à créer un rôle lié au service qui autorise IAM Access Analyzer à analyser les ressources dans d'autres services en votre nom. Cette autorisation autorise la création du rôle lié au service uniquement pour une utilisation par IAM Access Analyzer.
- AWS Organizations : autorise les administrateurs à utiliser IAM Access Analyzer pour une organisation dans AWS Organizations. Après avoir activé l'accès sécurisé pour IAM Access Analyzer dans AWS Organizations, les membres du compte de gestion peuvent consulter les résultats dans l'ensemble de leur organisation.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "access-analyzer:*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "access-analyzer.amazonaws.com"
        }
      }
```

```
    },
    {
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListChildren",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListParents",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

IAMAccessAnalyzerReadOnlyAccess

Utilisez la politique `IAMAccessAnalyzerReadOnlyAccess` AWS gérée pour autoriser l'accès en lecture seule à IAM Access Analyzer.

Pour autoriser également l'accès en lecture seule à IAM Access Analyzer pour AWS Organizations, créez une politique gérée par le client qui autorise les actions Décrire et Lister à partir de la stratégie gérée. IAMAccessAnalyzerFullAccess AWS

Service-level autorisations

Cette politique fournit un accès en lecture seule à IAM Access Analyzer. Cette politique n'inclut aucune autre autorisation de service.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Sid": "IAMAccessAnalyzerReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
        "access-analyzer:CheckAccessNotGranted",
        "access-analyzer:CheckNoNewAccess",
        "access-analyzer:Get*",
        "access-analyzer:List*",
        "access-analyzer:ValidatePolicy"
      ],
      "Resource": "*"
    }
  ]
}
```

AccessAnalyzerServiceRolePolicy

Vous ne pouvez pas vous connecter AccessAnalyzerServiceRolePolicy à vos entités IAM. Cette politique est attachée à un rôle lié à un service qui permet à l'analyseur d'accès IAM de réaliser des actions en votre nom. Pour plus d'informations, consultez la section Utilisation de rôles liés à des services pour Gestion des identités et des accès AWS et Access Analyzer.

Groupes d'autorisations

Cette politique autorise l'accès à IAM Access Analyzer pour analyser les métadonnées des ressources à partir de plusieurs Services AWS.

- Amazon DynamoDB : autorise l'affichage des flux et des tables DynamoDB.
- Amazon Elastic Compute Cloud : autorise la description des adresses IP, des instantanés et des VPC.
- Amazon Elastic Container Registry : autorise la description des référentiels d'images, la récupération des paramètres de compte et la récupération des politiques de registre et de référentiel.
- Amazon Elastic File System : autorise l'affichage de la description d'un système de fichiers Amazon EFS et de la politique de niveau de ressources d'un système de fichiers Amazon EFS.
- Gestion des identités et des accès AWS - Permet de récupérer des informations sur un rôle spécifié et de répertorier les rôles IAM dotés d'un préfixe de chemin spécifié. Permet de récupérer des informations sur les utilisateurs, les groupes IAM, les profils de connexion, les clés d'accès et les dernières données du service consultées.

- AWS Key Management Service - Autorise l'affichage d'informations détaillées sur une clé KMS, ses politiques clés et ses subventions.
- AWS Lambda - Autorise l'affichage des informations sur les alias, fonctions, couches et alias Lambda.
- AWS Organizations— Autorise AWS Organizations et autorise la création d'un analyseur au sein de l' AWS organisation en tant que zone de confiance.
- Amazon Relational Database Service : autorise l'affichage d'informations détaillées sur les instantanés de base de données Amazon RDS et les instantanés de clusters de bases de données Amazon RDS.
- Amazon Simple Storage Service : autorise l'affichage d'informations détaillées sur les points d'accès Amazon S3, les compartiments, les points d'accès aux compartiments de répertoire Amazon S3 et les compartiments de répertoire.
- AWS Secrets Manager - Autorise l'affichage d'informations détaillées sur les secrets et les politiques en matière de ressources associées aux secrets.
- Amazon Simple Notification Service : autorise l'affichage d'informations détaillées sur un sujet.
- Amazon Simple Queue Service : autorise l'affichage d'informations détaillées sur les files d'attente spécifiées.

Pour consulter la politique JSON, consultez AccessAnalyzerServiceRolePolicy le AWS Managed Policy Reference Guide.

IAMAuditRootUserCredentials

Utilisez la politique `IAMAuditRootUserCredentials` AWS gérée pour limiter les autorisations lorsque vous effectuez une tâche privilégiée sur un compte AWS Organizations membre afin d'auditer l'état des informations d'identification de l'utilisateur root d'un compte membre. Vous pouvez répertorier ou obtenir les informations d'identification de chaque utilisateur racine, notamment :

- S'il existe un mot de passe utilisateur racine
- Si l'utilisateur racine dispose d'une clé d'accès et quand elle a été utilisée pour la dernière fois
- Si l'utilisateur racine possède des certificats de signature associés
- Les dispositifs MFA associés à l'utilisateur racine
- Liste du statut consolidé des informations d'identification de l'utilisateur racine

Vous ne pouvez pas joindre de `IAMAuditRootUserCredentials` à vos entités IAM. Cette politique est associée AssumeRoot à l'exécution de tâches privilégiées sur un compte membre de votre organisation. Pour de plus amples informations, veuillez consulter Gestion centralisée de l'accès root pour les comptes membres.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "DenyAllOtherActionsOnAnyResource",
         "NotAction": [
            "iam:ListAccessKeys",
            "iam:ListSigningCertificates",
            "iam:GetLoginProfile",
            "iam:ListMFADevices",
            "iam:GetAccountSummary",
            "iam:GetUser",
            "iam:GetAccessKeyLastUsed"
         ],
         "Effect": "Deny",
         "Resource": "*"
      },
      {
         "Sid": "DenyAuditingCredentialsOnNonRootUserResource",
         "Action": [
            "iam:ListAccessKeys",
            "iam:ListSigningCertificates",
            "iam:GetLoginProfile",
            "iam:ListMFADevices" ,
            "iam:GetUser",
            "iam:GetAccessKeyLastUsed"
         ],
         "Effect": "Deny",
         "NotResource": "arn:aws:iam::*:root"
      }
   ]
}
```

Groupes d'autorisations

Cette politique est groupée en instructions basées sur le jeu d'autorisations fourni.

- DenyAllOtherActionsOnAnyResource— Refuse l'accès aux informations d'identification pour toutes les ressources.
- DenyAuditingCredentialsOnNonRootUserResource— Refuse l'accès aux informations d'identification pour toutes les ressources utilisateur non root.

IAMCreateRootUserPassword

Utilisez la politique `IAMCreateRootUserPassword` AWS gérée pour limiter les autorisations lorsque vous effectuez une tâche privilégiée sur un compte AWS Organizations membre afin de permettre la récupération du mot de passe d'un compte membre sans informations d'identification utilisateur root.

Vous ne pouvez pas joindre de `IAMCreateRootUserPassword` à vos entités IAM. Cette politique est associée AssumeRoot à l'exécution de tâches privilégiées sur un compte membre de votre organisation. Pour de plus amples informations, veuillez consulter Gestion centralisée de l'accès root pour les comptes membres.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "DenyAllOtherActionsOnAnyResource",
         "NotAction": [
            "iam:CreateLoginProfile",
            "iam:GetLoginProfile"
         ],
         "Effect": "Deny",
         "Resource": "*"
      },
      {
         "Sid": "DenyCreatingPasswordOnNonRootUserResource",
         "Action": [
            "iam:CreateLoginProfile",
```

```
            "iam:GetLoginProfile"
        ],
        "Effect": "Deny",
        "NotResource": "arn:aws:iam::*:root"
      }
   ]
}
```

Groupes d'autorisations

Cette politique est groupée en instructions basées sur le jeu d'autorisations fourni.

- DenyAllOtherActionsOnAnyResource— Refuse l'accès à l'obtention ou à la création d'un mot de passe pour toutes les ressources.
- DenyCreatingPasswordOnNonRootUserResource— Refuse l'accès à l'obtention ou à la création d'un mot de passe pour toutes les ressources utilisateur non root.

IAMDeleteRootUserCredentials

Utilisez la politique `IAMDeleteRootUserCredentials` AWS gérée pour limiter les autorisations lorsque vous effectuez une tâche privilégiée sur un compte AWS Organizations membre afin de supprimer les informations d'identification de l'utilisateur root, notamment le mot de passe, les clés d'accès, la signature de certificats et la désactivation de l'authentification multifacteur. Des autorisations supplémentaires sont requises pour cette action privilégiée, afin que vous puissiez afficher les dernières informations d'identification utilisées, vérifier les dernières informations utilisées pour le compte membre de l'utilisateur racine et dresser la liste des autorisations pour toutes les informations d'identification de l'utilisateur racine à supprimer.

Vous ne pouvez pas joindre de `IAMDeleteRootUserCredentials` à vos entités IAM. Cette politique est associée AssumeRoot à l'exécution de tâches privilégiées sur un compte membre de votre organisation. Pour de plus amples informations, veuillez consulter Gestion centralisée de l'accès root pour les comptes membres.

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
```

```
  {
   "Sid": "DenyAllOtherActionsOnAnyResource",
   "Effect": "Deny",
   "NotAction": [
    "iam:DeleteAccessKey",
    "iam:DeleteSigningCertificate",
    "iam:DeleteLoginProfile",
    "iam:DeactivateMFADevice",
    "iam:ListAccessKeys",
    "iam:ListSigningCertificates",
    "iam:GetLoginProfile",
    "iam:ListMFADevices",
    "iam:GetUser",
    "iam:GetAccessKeyLastUsed"
   ],

   "Resource": "*"
  },
  {
   "Sid": "DenyDeletingRootUserCredentialsOnNonRootUserResource",
   "Effect": "Deny",
   "Action": [
    "iam:DeleteAccessKey",
    "iam:DeleteSigningCertificate",
    "iam:DeleteLoginProfile",
    "iam:DeactivateMFADevice",
    "iam:ListAccessKeys",
    "iam:ListSigningCertificates",
    "iam:GetLoginProfile",
    "iam:ListMFADevices",
    "iam:GetUser",
    "iam:GetAccessKeyLastUsed"
   ],

   "NotResource": "arn:aws:iam::*:root"
  }
 ]
}
```

Groupes d'autorisations

Cette politique est groupée en instructions basées sur le jeu d'autorisations fourni.

- DenyAllOtherActionsOnAnyResource— Refuse l'accès à l'obtention ou à la suppression des informations d'identification pour toutes les ressources.
- DenyDeletingRootUserCredentialsOnNonRootUserResource— Refuse l'accès à l'obtention ou à la suppression d'informations d'identification pour toutes les ressources utilisateur non root.

S3UnlockBucketPolicy

Utilisez la politique `S3UnlockBucketPolicy` AWS gérée pour limiter les autorisations lorsque vous effectuez une tâche privilégiée sur un compte AWS Organizations membre afin de supprimer une politique de compartiment mal configurée qui empêche tous les principaux utilisateurs d'accéder à un compartiment Amazon S3.

Vous ne pouvez pas joindre de `S3UnlockBucketPolicy` à vos entités IAM. Cette politique est associée AssumeRoot à l'exécution de tâches privilégiées sur un compte membre de votre organisation. Pour de plus amples informations, veuillez consulter Gestion centralisée de l'accès root pour les comptes membres.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "DenyAllOtherActionsOnAnyResource",
         "NotAction": [
            "s3:DeleteBucketPolicy",
            "s3:PutBucketPolicy",
            "s3:GetBucketPolicy",
            "s3:ListAllMyBuckets"
         ],
         "Effect": "Deny",
         "Resource": "*"
      },
      {
         "Sid": "DenyManagingBucketPolicyForNonRootCallers",
         "Action": [
            "s3:DeleteBucketPolicy",
            "s3:PutBucketPolicy",
            "s3:GetBucketPolicy",
            "s3:ListAllMyBuckets"
```

```
        ],
        "Effect": "Deny",
        "Resource": "*",
        "Condition" : {
           "ArnNotLike" : {
              "aws:PrincipalArn" : "arn:aws:iam::*:root"
           }
        }
     }
  ]
}
```

Groupes d'autorisations

Cette politique est groupée en instructions basées sur le jeu d'autorisations fourni.

- DenyAllOtherActionsOnAnyResource— Refuse l'accès aux politiques relatives aux compartiments pour toutes les ressources.
- DenyManagingBucketPolicyForNonRootCallers— Refuse l'accès aux politiques de bucket pour toutes les ressources utilisateur non root.

SQSUnlockQueuePolicy

Utilisez la politique `SQSUnlockQueuePolicy` AWS gérée pour limiter les autorisations lorsque vous effectuez une tâche privilégiée sur un compte AWS Organizations membre afin de supprimer une politique basée sur les ressources Amazon Simple Queue Service qui interdit à tous les mandants d'accéder à une file d'attente Amazon SQS.

Vous ne pouvez pas joindre de `SQSUnlockQueuePolicy` à vos entités IAM. Cette politique est associée AssumeRoot à l'exécution de tâches privilégiées sur un compte membre de votre organisation. Pour de plus amples informations, veuillez consulter Gestion centralisée de l'accès root pour les comptes membres.

Groupes d'autorisations

Cette politique est groupée en instructions basées sur le jeu d'autorisations fourni.

- DenyAllOtherActionsOnAnyResource— Refuse l'accès aux actions Amazon SQS pour toutes les ressources.

- DenyGettingQueueAttributesOnNonOwnQueue— Refuse l'accès aux attributs de file d'attente Amazon SQS pour les files d'attente appartenant à un autre compte.
- DenyActionsForNonRootUser— Refuse l'accès aux actions Amazon SQS pour toutes les ressources utilisateur non root.

AccountAccessManagerServiceRolePolicy

Vous ne pouvez pas joindre de `AccountAccessManagerServiceRolePolicy` à vos entités IAM. Cette politique est associée à un rôle lié à un service qui permet au gestionnaire d'accès au compte d'effectuer des actions en votre nom. Pour de plus amples informations, veuillez consulter the section called "Sécurité".

Service-level autorisations

Cette politique permet au gestionnaire d'accès aux comptes de récupérer des informations sur les comptes de votre organisation afin de maintenir une composition précise des comptes à des fins de gestion des accès. Le gestionnaire d'accès aux comptes utilise ces autorisations pour détecter quand les comptes quittent votre organisation, vérifier si l'accès aux services sécurisés est activé et confirmer l'état du compte avant de gérer les droits.

- AWS Organizations— Autorise les autorisations nécessaires pour répertorier les comptes de l'organisation, vérifier si l'accès aux services sécurisés est activé et décrire l'état du compte.

Pour consulter la politique JSON, consultez AccountAccessManagerServiceRolePolicy le AWS Managed Policy Reference Guide.

Mises à jour d'IAM, d'IAM Access Analyzer et du gestionnaire d'accès aux comptes pour AWS stratégies gérées

Consultez les détails des mises à jour apportées à IAM et aux politiques AWS gérées depuis que le service a commencé à suivre ces modifications. Pour recevoir des alertes automatiques sur les modifications apportées à cette page, abonnez-vous au flux RSS sur les pages d'historique d'IAM et IAM Access Analyzer Document.

Modifier	Description	Date
AccountAccessManagerServiceRolePolicy — Ajout d'une politique gérée	Le gestionnaire d'accès aux comptes a ajouté une politique de rôle lié au service qui permet au service de récupérer des informations sur les comptes de votre organisation.	31 juillet 2026
IAMUserChangePassword : autorisations ajoutées	IAM a ajouté des autorisations à `IAMUserChangePassword` pour autoriser les utilisateurs spécifiés dans un chemin.	28 mai 2025
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté `iam:GetAccountAuthorizationDetails` sur les autorisations de niveau de service de `AccessAnalyzerServiceRolePolicy` .	12 mai 2025
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté la prise en charge des points d'accès aux compartiments de répertoires Amazon S3 aux autorisations de niveau de service de `AccessAnalyzerServiceRolePolicy` .	31 mars 2025
IAMDeleteRootUserCredentials – Supprimer des autorisations	IAM a supprimé l'autorisation `iam:DeleteVirtualMFADevice` de la politique gérée.	7 janvier 2025

Modifier	Description	Date
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté la prise en charge de l'autorisation de récupérer des informations sur les paramètres du compte Amazon ECR et les politiques de registre aux autorisations de niveau de service de `AccessAnalyzerServiceRolePolicy` .	10 décembre 2024
IAMAuditRootUserCredentials — Ajout d'une politique gérée	IAM a ajouté des politiques gérées pour gérer de manière centralisée l'accès racine aux comptes membres afin de délimiter les tâches privilégiées que vous pouvez effectuer sur les comptes membres AWS Organizations .	14 novembre 2024
IAMCreateRootUserPassword— Ajout d'une politique gérée	IAM a ajouté des politiques gérées pour gérer de manière centralisée l'accès racine aux comptes membres afin de délimiter les tâches privilégiées que vous pouvez effectuer sur les comptes membres AWS Organizations .	14 novembre 2024

Modifier	Description	Date
IAMDeleteRootUserCredentials— Ajout d'une politique gérée	IAM a ajouté des politiques gérées pour gérer de manière centralisée l'accès racine aux comptes membres afin de délimiter les tâches privilégiées que vous pouvez effectuer sur les comptes membres AWS Organizations .	14 novembre 2024
S3UnlockBucketPolicy— Ajout d'une politique gérée	IAM a ajouté des politiques gérées pour gérer de manière centralisée l'accès racine aux comptes membres afin de délimiter les tâches privilégiées que vous pouvez effectuer sur les comptes membres AWS Organizations .	14 novembre 2024
SQSUnlockQueuePolicy— Ajout d'une politique gérée	IAM a ajouté des politiques gérées pour gérer de manière centralisée l'accès racine aux comptes membres afin de délimiter les tâches privilégiées que vous pouvez effectuer sur les comptes membres AWS Organizations .	14 novembre 2024

Modifier	Description	Date
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté la prise en charge de l'autorisation de récupérer des informations sur les balises de rôle et d'utilisateur IAM aux autorisations de niveau de service de `AccessAnalyzerServiceRolePolicy` .	29 octobre 2024
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté la prise en charge de l'autorisation de récupérer des informations sur les politiques de rôle et d'utilisateur IAM aux autorisations de niveau de service de `AccessAnalyzerServiceRolePolicy` .	30 mai 2024
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté la prise en charge de l'accès public des instantanés Amazon EC2 aux autorisations de niveau de service de `AccessAnalyzerServiceRolePolicy` .	23 janvier 2024
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté la prise en charge des flux et des tables DynamoDB aux autorisations de niveau de service de `AccessAnalyzerServiceRolePolicy` .	11 janvier 2024

Modifier	Description	Date
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté la prise en charge des compartiements de répertoires Amazon S3 aux autorisations de niveau de service de `AccessAnalyzerServiceRolePolicy` .	1er décembre 2023
IAMAccessAnalyzerReadOnlyAccess : autorisations ajoutées	L'analyseur d'accès IAM a ajouté des autorisations pour vous permettre de vérifier si les mises à jour de vos politiques accordent un accès supplémentaire. L'analyseur d'accès IAM a besoin de cette autorisation pour effectuer des vérifications de politique sur vos politiques.	26 novembre 2023
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté des actions IAM aux autorisations de niveau du service d'`AccessAnalyzerServiceRolePolicy` pour prendre en charge les actions suivantes : • Lister les entités pour une politique • Générer les dernières informations consultées sur le service • Lister les informations clés d'accès	26 novembre 2023

Modifier	Description	Date
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	IAM Access Analyzer a ajouté la prise en charge des types de ressources suivants sur les autorisations de niveau de service de `AccessAnalyzerServiceRolePolicy` : • Instantanés de volumes Amazon EBS • Référentiels Amazon ECR • Système de fichiers Amazon EFS • Instantanés de base de données Amazon RDS • Instantanés du cluster de base de données Amazon RDS • Rubriques Amazon SNS	25 octobre 2022
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté l'action `lambda:GetFunctionUrlConfig` sur les autorisations de niveau de service de `AccessAnalyzerServiceRolePolicy` .	6 avril 2022
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté de nouvelles actions Amazon S3 pour analyser les métadonnées associées aux points d'accès multi-régions.	2 septembre 2021

Modifier	Description	Date
IAMAccessAnalyzerReadOnlyAccess : autorisations ajoutées	L'analyseur d'accès IAM a ajouté une nouvelle action pour octroyer des autorisations `ValidatePolicy` visant à vous autoriser à utiliser les vérifications de politique pour la validation. L'analyseur d'accès IAM a besoin de cette autorisation pour effectuer des vérifications de politique sur vos politiques.	16 mars 2021
IAM Access Analyzer a commencé à suivre les modifications	IAM Access Analyzer a commencé à suivre les modifications apportées à ses politiques AWS gérées.	1er mars 2021

Fonctions de sécurité en dehors d'IAM

Vous utilisez IAM pour contrôler l'accès aux tâches effectuées à l'aide des outils de Console de gestion AWS ligne de AWS commande ou des opérations d'API de service à l'aide du AWS SDKs. Certains produits AWS offrent également d'autres moyens de sécuriser leurs ressources. La liste suivante fournit quelques exemples, mais elle n'est pas exhaustive.

Amazon EC2

Dans Amazon Elastic Compute Cloud, vous vous connectez à une instance à l'aide d'une paire de clés (pour les instances Linux) ou d'un nom utilisateur et mot de passe (pour les instances Microsoft Windows).

Pour plus d'informations, consultez la documentation suivante :

- Commencer à utiliser les instances Amazon EC2 Linux dans le guide de EC2 l'utilisateur Amazon

- Commencer à utiliser les instances Amazon EC2 Windows dans le guide de EC2 l'utilisateur Amazon

Amazon RDS

Dans Amazon Relational Database Service, vous vous connectez au moteur de base de données à l'aide d'un nom utilisateur et mot de passe qui sont associés à la base de données.

Pour de plus amples informations, veuillez consulter Mise en route avec Amazon RDS dans le Guide de l'utilisateur Amazon RDS.

Amazon EC2 et Amazon RDS

Dans Amazon EC2 et Amazon RDS, vous utilisez des groupes de sécurité pour contrôler le trafic vers une instance ou une base de données.

Pour plus d'informations, consultez la documentation suivante :

- Groupes EC2 de sécurité Amazon pour instances Linux dans le guide de EC2 l'utilisateur Amazon
- Groupes EC2 de sécurité Amazon pour les instances Windows dans le guide de EC2 l'utilisateur Amazon
- Groupes de sécurité Amazon RDS dans le Guide de l'utilisateur Amazon RDS

WorkSpaces

Sur Amazon WorkSpaces, les utilisateurs se connectent à un ordinateur de bureau à l'aide d'un nom d'utilisateur et d'un mot de passe.

Pour plus d'informations, consultez Getting Started with WorkSpaces dans le guide d' WorkSpaces administration Amazon.

Ces méthodes de contrôle d'accès ne font pas partie d'IAM. IAM vous permet de contrôler la façon dont ces AWS produits sont administrés : création ou résiliation d'une EC2 instance Amazon, configuration de nouveaux WorkSpaces bureaux, etc. Autrement dit, IAM vous aide à contrôler les tâches exécutées en effectuant des demandes à Amazon Web Services ; vous pouvez également contrôler l'accès à Console de gestion AWS. Cependant, IAM ne vous aide pas à gérer la sécurité pour des tâches telles que la connexion à un système d'exploitation (Amazon EC2), à une base de données (Amazon RDS) ou à un ordinateur de bureau (Amazon WorkSpaces).

Lorsque vous travaillez avec un AWS produit spécifique, assurez-vous de lire la documentation pour connaître les options de sécurité pour toutes les ressources appartenant à ce produit.

Utilisation AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer fournit les fonctionnalités suivantes :

- Les analyseurs d'accès externe de l'analyseur d'accès IAM vous aide à identifier les ressources de votre organisation et de vos comptes qui sont partagés avec une entité externe.
- Les analyseurs d'accès internes de l'analyseur d'accès IAM aident à identifier les accès internes à certaines ressources au sein de votre organisation et de vos comptes.
- Les analyseurs d'accès non utilisés de l'analyseur d'accès IAM aident à identifier les accès non utilisés au sein de votre organisation et de vos comptes.
- IAM Access Analyzer valide les politiques IAM par rapport à la grammaire des politiques et aux meilleures pratiques. AWS
- Les vérifications de politiques personnalisées de l'analyseur d'accès IAM permettent de valider les politiques IAM par rapport aux normes de sécurité spécifiées..
- IAM Access Analyzer génère des politiques IAM en fonction de l'activité d'accès dans vos journaux. AWS CloudTrail

Identification des ressources partagées avec une entité externe

L'IAM Access Analyzer vous aide à identifier les ressources de votre organisation et de vos comptes, telles que les compartiments Amazon S3 ou les rôles IAM, partagés avec une entité externe. Cela vous permet d'identifier les accès imprévus à vos ressources et données, ce qui constitue un risque de sécurité. IAM Access Analyzer identifie les ressources partagées avec des donneurs d'ordre externes en utilisant un raisonnement logique pour analyser les politiques basées sur les ressources dans votre environnement. AWS Pour chaque instance d'une ressource qui est partagée en dehors de votre compte, l'IAM Access Analyzer génère un résultat. Les résultats comprennent des renseignements sur l'accès et le principal externe à qui il est accordé. Vous pouvez réviser les résultats pour déterminer si l'accès est intentionnel et sûr ou s'il est non intentionnel et représente un risque pour la sécurité. En plus de vous aider à identifier les ressources partagées avec une entité externe, vous pouvez utiliser les résultats de l'IAM Access Analyzer pour prévisualiser la façon dont votre politique affecte l'accès public et les entre comptes à votre ressource avant de déployer des autorisations de ressources. Les résultats sont organisés dans un tableau de bord récapitulatif visuel. Le tableau de bord met en évidence la distinction entre les résultats relatifs à l'accès public et ceux

relatifs à l'accès intercomptes, et fournit une ventilation des résultats par type de ressource. Pour plus d'informations sur les tableaux de bord, consultez Affichage du tableau de bord des résultats de l'analyseur d'accès IAM.

> **Note**
>
> Une entité externe peut être un autre AWS compte, un utilisateur root, un utilisateur ou un rôle IAM, un utilisateur fédéré, un utilisateur anonyme ou une autre entité que vous pouvez utiliser pour créer un filtre. Pour plus d'informations, veuillez consulter Éléments de politique JSON AWS : Principal.

Pour obtenir des instructions sur la façon de démarrer, consultezCréation d'un analyseur d'accès IAM pour l'analyse de l'accès externe. Une fois que vous avez créé un analyseur, c'est possibleGestion d'un analyseur d'accès IAM pour l'analyse de l'accès externe.

Lorsque vous activez l'IAM Access Analyzer, vous créez un analyseur pour l'ensemble de votre organisation ou de votre compte. L'organisation ou le compte que vous sélectionnez est la zone de confiance de l'analyseur. L'analyseur surveille toutes les ressources prises en charge dans votre zone de confiance. Tout accès aux ressources par les principaux qui se trouvent dans votre zone de confiance est considéré comme fiable. Une fois activé, l'IAM Access Analyzer analyse les politiques appliquées à toutes les ressources prises en charge dans votre zone de confiance. Après la première analyse, l'IAM Access Analyzer analyse lesdites politiques de façon périodique. Si vous ajoutez une nouvelle politique ou modifiez une déja existante, l'analyseur d'accès IAM analyse la nouvelle ou celle mise à jour dans un délai d'environ 30 minutes.

Lors de l'analyse des politiques, si l'IAM Access Analyzer en identifie une qui accorde l'accès à un principal externe ne se trouvant pas dans votre zone de confiance, il génère un résultat. Chaque résultat inclut des détails sur la ressource, l'entité externe qui y a accès et les autorisations accordées, afin que vous puissiez prendre les mesures appropriées. Vous pouvez afficher les détails inclus dans le résultat pour déterminer si l'accès à la ressource est intentionnel ou représente un risque potentiel que vous devez résoudre. Lorsque vous ajoutez une politique à une ressource ou que mettez à jour une politique existante, l'IAM Access Analyzer. En outre, l'IAM Access Analyzer analyse périodiquement toutes les politiques basées sur les ressources.

Dans de rares cas et sous certaines conditions, l'analyseur d'accès IAM ne reçoit pas de notification d'ajout ou de mise à jour d'une politique, ce qui peut entraîner des retards dans les résultats générés. Il peut falloir jusqu'à 6 heures à l'analyseur d'accès IAM pour générer ou résoudre des résultats si

vous créez ou supprimez un point d'accès multi-régions associé à un compartiment Amazon S3, ou si vous mettez à jour la politique pour le point d'accès multi-régions. De plus, en cas de problème de livraison du AWS CloudTrail journal ou de modification des restrictions de la politique de contrôle des ressources (RCP), le changement de politique ne déclenche pas une nouvelle analyse de la ressource signalée dans le résultat. Lorsque cela se produit, l'IAM Access Analyzer analyse la politique nouvelle ou mise à jour au cours de l'analyse périodique suivante, qui a lieu dans un délai maximal de 24 heures. Si vous souhaitez confirmer qu'une modification apportée à une politique résout un problème d'accès signalé dans un résultat, vous pouvez réanalyser la ressource indiquée dans le résultat à l'aide du lien Réanalyser de la page des détails Résultats ou à l'aide de l'opération `StartResourceScan` de l'API de l'analyseur d'accès IAM. Pour en savoir plus, veuillez consulter la section Résolution des problèmes liés aux résultats de l'analyseur d'accès IAM.

> **Important**
>
> Pour un accès externe, l'analyseur d'accès IAM analyse uniquement les politiques appliquées aux ressources de la même région AWS dans laquelle il est activé. Pour surveiller toutes les ressources de votre AWS environnement, vous devez créer un analyseur d'accès externe afin d'activer IAM Access Analyzer dans chaque région où vous utilisez des ressources prises en charge. AWS
>
> En cas d'accès non utilisé, les résultats de l'analyseur ne changent pas en fonction de la région. Il n'est pas nécessaire de créer un analyseur d'accès non utilisé dans chaque région où vous disposez de ressources.

L'analyseur d'accès IAM prend en charge les types de ressources suivants pour l'accès externe :

- Compartiments Amazon Simple Storage Service
- Compartiments du répertoire Amazon Simple Storage Service
- Gestion des identités et des accès AWS rôles
- AWS Key Management Service clés
- AWS Lambda fonctions et couches
- Files d'attente Amazon Simple Queue Service
- AWS Secrets Manager secrets
- Rubriques Amazon Simple Notification Service
- Instantanés volumes Amazon Elastic Block Store
- Instantanés de base de données service de base de données relationnelle Amazon

- Instantanés de cluster de base de données service base de données relationnelle Amazon
- Référentiels Amazon Elastic Container Registry
- Systèmes de fichiers Amazon Elastic File System
- Amazon DynamoDB Streams
- Tables Amazon DynamoDB

Identification des accès internes aux ressources critiques pour l'entreprise

Pour les ressources critiques que vous avez sélectionnées, l'analyseur d'accès IAM vous aide à identifier les principaux au sein de votre organisation ou de votre compte qui y ont accès. Cette analyse soutient la mise en œuvre du principe du moindre privilège en garantissant que vos ressources spécifiées ne soient accessibles qu'aux principaux intéressés au sein de votre organisation.

Pour obtenir des instructions sur la façon de démarrer, consultezCréation d'un analyseur d'accès IAM pour l'analyse de l'accès interne. Une fois que vous avez créé un analyseur, c'est possibleGestion d'un analyseur d'accès IAM pour l'analyse de l'accès interne.

L'analyse des accès internes vous aide à :

- Déterminer quels utilisateurs ou rôles IAM au sein de votre compte ou de votre organisation peuvent accéder aux ressources que vous avez spécifiées
- Comprendre les chemins d'accès entre les principaux et les ressources au sein de votre environnement AWS
- Vérifier que vos contrôles d'accès fonctionnent comme prévu
- Examiner les résultats afin de déterminer si l'accès est intentionnel et sécurisé ou s'il est involontaire et présente un risque pour la sécurité
- Identifier et remédier aux accès non autorisés au sein de votre organisation

L'analyseur d'accès IAM prend en charge les types de ressources suivants pour l'accès interne :

- Compartiments Amazon Simple Storage Service
- Compartiments du répertoire Amazon Simple Storage Service
- Instantanés de base de données service de base de données relationnelle Amazon

- Instantanés de cluster de base de données service base de données relationnelle Amazon
- Amazon DynamoDB Streams
- Tables Amazon DynamoDB

Identification des accès non utilisés accordés aux utilisateurs et aux rôles IAM

IAM Access Analyzer vous aide à identifier et à examiner les accès non utilisés dans votre AWS organisation et vos comptes. L'analyseur d'accès IAM surveille en permanence tous les rôles et utilisateurs IAM dans votre organisation et vos comptes AWS et génère des résultats pour les accès non utilisés. Les résultats mettent en évidence les rôles non utilisés ainsi que les clés d'accès et les mots de passe non utilisés pour les utilisateurs IAM. Pour les rôles et les utilisateurs IAM actifs, les résultats fournissent une visibilité sur les services et les actions non utilisés.

Pour obtenir des instructions sur la création d'un analyseur d'accès non utilisé et la gestion de ses résultats, consultezGestion d'un analyseur d'accès IAM pour l'analyse de l'accès non utilisé.

IAM Access Analyzer passe en revue les dernières informations consultées pour tous les rôles et comptes de votre AWS organisation afin de vous aider à identifier les accès non utilisés. Les informations auxquelles vous avez accédé pour la dernière fois sur les actions IAM vous aident à identifier les actions non utilisées pour les rôles de vos Comptes AWS. Pour de plus amples informations, veuillez consulter Affiner les autorisations en AWS utilisant les dernières informations consultées.

Les résultats relatifs aux analyseurs d'accès externes, internes et non utilisés sont organisés dans un tableau de bord récapitulatif visuel. Le tableau de bord met en évidence vos AWS ressources et celles Comptes AWS qui contiennent le plus de résultats et fournit une ventilation des résultats par type. Pour plus d'informations sur le tableau de bord, consultez Affichage du tableau de bord des résultats de l'analyseur d'accès IAM.

Validation des politiques par rapport à AWS bonnes pratiques

Vous pouvez valider vos politiques par rapport à la grammaire des politiques IAM et aux bonnes pratiques AWS à l'aide des vérifications de politique de base fournies par la validation des politiques de l'analyseur d'accès IAM. Vous pouvez créer ou modifier une politique à l'aide de l' AWS CLI AWS API ou de l'éditeur de règles JSON de la console IAM. Vous pouvez afficher les résultats des

vérifications de validation de politique qui incluent des avertissements de sécurité, des erreurs, des avertissements généraux et des suggestions pour votre politique. Ces résultats fournissent des recommandations pratiques qui vous aident à élaborer des politiques fonctionnelles et conformes aux AWS meilleures pratiques. Pour en savoir plus sur la validation des politiques à l'aide de la validation des politiques, consultez Validation des politiques à l'aide de l'analyseur d'accès IAM. Pour obtenir la liste des contrôles de politique effectués par IAM Access Analyzer, consultez. Vérifications de validation des politiques

Validation des politiques par rapport aux normes de sécurité que vous avez spécifiées

Vous pouvez valider vos politiques par rapport aux normes de sécurité spécifiées à l'aide des vérifications de politiques personnalisées de l'analyseur d'accès IAM. Vous pouvez créer ou modifier une politique à l'aide de l' AWS CLI AWS API ou de l'éditeur de règles JSON de la console IAM. Avec la console, vous pouvez vérifier si votre politique mise à jour accorde un nouvel accès par rapport à la version existante. Grâce à AWS CLI une AWS API, vous pouvez également vérifier que les actions IAM spécifiques que vous considérez comme critiques ne sont pas autorisées par une politique. Ces vérifications mettent en évidence une déclaration de politique qui accorde un nouvel accès. Vous pouvez mettre à jour la déclaration de politique et réexécuter les vérifications jusqu'à ce que la politique soit conforme à votre norme de sécurité. Pour en savoir plus sur la validation des politiques à l'aide de vérifications de politique personnalisées, consultez Validation des politiques à l'aide des vérifications des politiques personnalisées de l'analyseur d'accès IAM.

Générer des politiques

IAM Access Analyzer analyse vos AWS CloudTrail journaux pour identifier les actions et les services qui ont été utilisés par une entité IAM (utilisateur ou rôle) au cours de la plage de dates que vous avez spécifiée. Il génère ensuite une politique IAM basée sur cette activité d'accès. Vous pouvez utiliser la politique générée pour affiner les autorisations d'une entité, en l'attachant à un utilisateur ou un rôle IAM. Pour en savoir plus sur la génération de politiques à l'aide de l'IAM Access Analyzer, veuillez consulter Génération d'une politique IAM Access Analyzer.

Tarification pour l'analyseur d'accès IAM

L'analyseur d'accès IAM facture l'analyse des accès non utilisés en fonction du nombre de rôles et d'utilisateurs IAM analysés par analyseur et par mois.

- Vous serez facturé pour chaque analyseur d’accès non utilisé que vous créez.
- Si vous créez des analyseurs d’accès non utilisés dans plusieurs régions, vous serez facturé pour chaque analyseur.
- Service-linked les rôles ne sont pas analysés pour détecter les activités d'accès inutilisées et ils ne sont pas inclus dans le nombre total de rôles IAM analysés.

L’analyseur d’accès IAM facture l’analyse des accès internes en fonction du nombre de ressources surveillées par analyseur d’accès interne et par mois.

L’analyseur d’accès IAM facture les vérifications de politiques personnalisées en fonction du nombre de demandes d’API faites à l’analyseur d’accès IAM pour vérifier les nouveaux accès.

Pour obtenir la liste complète des prix et des frais associés à IAM Access Analyzer, consultez la rubrique Tarification d'IAM Access Analyzer.

Pour consulter votre facture, dirigez-vous vers le Tableau de bord de gestion des coûts et de la facturation dans la console AWS Billing and Cost Management. Votre facture contient des liens vers les rapports d’utilisation qui fournissent des détails sur votre facture. Pour en savoir plus sur Compte AWS la facturation, consultez le guide de AWS Billing l'utilisateur

Si vous avez des questions concernant AWS la facturation, les comptes et les événements, contactez Support.

Résultats de l’analyseur d’accès IAM

IAM Access Analyzer génère des résultats concernant les accès externes, les accès internes et les accès non utilisés au sein de votre organisation Compte AWS ou de votre organisation.

Pour l’accès externe, l’analyseur d’accès IAM génère un résultat pour chaque instance de politique basée sur les ressources accordant l’accès à une ressource dans votre zone de confiance à un principal qui ne se trouve dans votre zone de confiance. Lorsque vous créez un analyseur d'accès externe, vous choisissez une organisation ou vous souhaitez l' Compte AWS analyser. Tout principal de l'organisation ou du compte que vous sélectionnez pour l'analyseur est considéré comme approuvé. Étant donné que les principaux de la même organisation ou du même compte sont approuvés, les ressources et les principaux de l'organisation ou du compte constituent la zone de confiance de l'analyseur. Tout partage se trouvant dans la zone de confiance est considéré comme sûr. L’IAM Access Analyzer ne génère donc pas de résultat. Par exemple, si vous sélectionnez une organisation comme zone de confiance d'un analyseur, toutes les ressources et tous les principaux

de cette organisation sont compris dans cette zone de confiance. Si vous accordez des autorisations à un compartiment Amazon S3 dans l'un des comptes membres de votre organisation à un principal dans un autre compte membre de celle-ci, l'analyseur d'accès IAM ne génère pas de résultat. Toutefois, si vous accordez une autorisation à un principal dans un compte qui n'est pas membre de l'organisation, l'IAM Access Analyzer génère un résultat.

Pour les accès internes, l'analyseur d'accès IAM génère des résultats lorsqu'il existe un chemin d'accès possible entre un rôle ou un utilisateur IAM au sein de votre organisation et de vos ressources spécifiées. Tout comme pour l'analyse des accès externes, la portée que vous choisissez (organisation ou compte) détermine ce qui est considéré comme interne. Si vous sélectionnez une organisation comme portée, l'analyseur d'accès IAM générera des résultats pour les chemins d'accès entre les principaux et les ressources au sein de votre organisation. Si vous sélectionnez un compte, les résultats seront générés pour les chemins d'accès au sein de ce compte spécifique. L'analyseur d'accès IAM utilise un raisonnement automatisé pour évaluer toutes les politiques IAM afin de contrôler qui a accès à vos ressources.

La combinaison des résultats d'accès externes et internes avec la même zone de confiance fournit une analyse complète de tous les accès possibles à une ressource particulière, tant à l'intérieur qu'à l'extérieur de votre périmètre de confiance défini.

Pour les accès non utilisés, l'analyseur d'accès IAM génère des résultats concernant les accès non utilisés accordés dans votre organisation et vos comptes AWS . Lorsque vous créez un analyseur d'accès non utilisé, IAM Access Analyzer surveille en permanence tous les rôles et utilisateurs IAM de votre AWS organisation et de vos comptes et génère des résultats pour les accès non utilisés. L'analyseur d'accès IAM génère les types de résultats suivants pour les accès non utilisés :

- Rôles non utilisés : rôles sans activité d'accès dans la fenêtre d'utilisation spécifiée.
- Clés d'accès et mots de passe utilisateur IAM non utilisés : informations d'identification appartenant aux utilisateurs IAM qui n'ont pas été utilisées pour accéder à votre compte Compte AWS dans la fenêtre d'utilisation spécifiée.
- Autorisations non utilisées : autorisations au niveau du service et de l'action qui n'ont pas été utilisées par un rôle dans la fenêtre d'utilisation spécifiée. L'analyseur d'accès IAM utilise des politiques basées sur l'identité attachées aux rôles pour déterminer les services et les actions auxquels ces rôles peuvent accéder. L'analyseur d'accès IAM prend en charge l'examen des autorisations non utilisées pour toutes les autorisations de niveau du service. Pour obtenir la liste complète des autorisations au niveau de l'action prises en charge pour les résultats des accès non utilisés, consultez Services et actions concernant les dernières informations consultées relatives à une action IAM.

> **Note**
>
> L'analyseur d'accès IAM fournit gratuitement des résultats d'analyse des accès externes. Des frais sont facturés pour les résultats d'accès inutilisés en fonction du nombre de rôles IAM et d'utilisateurs analysés par analyseur et par mois. Des frais sont également facturés pour les résultats d'accès internes basés sur le nombre de AWS ressources surveillées par analyseur et par mois. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

Rubriques

- Fonctionnement des résultats de l'analyseur d'accès IAM
- Démarrage avec AWS Identity and Access Management Access Analyzer
- Affichage du tableau de bord des résultats de l'analyseur d'accès IAM
- Examen des résultats de l'analyseur d'accès IAM
- Filtrage des résultats de l'analyseur d'accès IAM
- Archivage des résultats de l'analyseur d'accès IAM
- Résolution des problèmes liés aux résultats de l'analyseur d'accès IAM
- Résultats d'erreur de l'analyseur d'accès IAM
- Types de ressources prises en charge par l'analyseur d'accès IAM pour l'accès externe et interne
- Administrateur délégué pour IAM Access Analyzer
- Règles d'archivage
- Surveillance AWS Identity and Access Management Access Analyzer avec Amazon EventBridge
- Intégrez IAM Access Analyzer à AWS Security Hub CSPM
- Journalisation des appels d'API IAM Access Analyzer avec AWS CloudTrail
- Clés de filtre de l'IAM Access Analyzer
- Utilisation de rôles liés à un service pour AWS Identity and Access Management Access Analyzer

Fonctionnement des résultats de l'analyseur d'accès IAM

Cette rubrique décrit les concepts et les termes utilisés dans IAM Access Analyzer pour vous aider à vous familiariser avec la manière dont IAM Access Analyzer surveille l'accès à vos ressources. AWS

Résultats des accès externes

Les résultats d'accès externe sont générés une seule fois pour chaque instance d'une ressource qui est partagée en dehors de votre zone de confiance. Chaque fois qu'une politique basée sur les ressources est modifiée, IAM Access Analyzer analyse la politique. Si la politique mise à jour partage une ressource déjà identifiée dans un résultat, mais avec des autorisations ou des conditions différentes, un nouveau résultat est généré pour cette instance du partage de ressource. Les modifications apportées à une politique de contrôle des ressources qui ont une incidence sur la restriction de la politique de contrôle des ressources (RCP) génèrent également un nouveau résultat. L'analyseur d'accès IAM évalue également les configurations de contrôle d'accès établies par des politiques déclaratives. Si l'accès dans le premier résultat est supprimé, celui-ci est mis à jour et prend le statut Résolu.

Le statut de tous les résultats reste Actif jusqu'à ce que vous les archiviez ou que vous supprimiez l'accès ayant généré le résultat. Lorsque vous supprimez l'accès, le statut du résultat est mis à jour et devient Résolu.

> **Note**
>
> Lorsque la politique est modifiée, l'analyseur d'accès IAM peut prendre jusqu'à 30 minutes pour analyser la ressource et mettre à jour le résultat d'accès externe. Les modifications apportées à une politique de contrôle des ressources (RCP) n'entraînent pas une nouvelle analyse de la ressource mentionnée dans le résultat. L'analyseur d'accès IAM analyse la politique nouvelle ou mise à jour au cours de l'analyse périodique suivante, qui a lieu dans un délai maximal de 24 heures.

Processus de génération de résultats relatifs aux accès externes par l'analyseur d'accès IAM

AWS Identity and Access Management Access Analyzer utilise une technologie appelée Zelkova pour analyser les politiques IAM et identifier les accès externes aux ressources.

Zelkova traduit les politiques IAM en instructions logiques équivalentes et les soumet à un ensemble de solveurs logiques généraux et spécialisés (théories du module de satisfiabilité). L'analyseur d'accès IAM applique Zelkova de manière répétitive à une politique, en utilisant des requêtes de plus en plus spécifiques pour caractériser les types d'accès que la politique autorise en fonction de

son contenu. Pour en savoir plus sur les théories du module de satisfiabilité, consultez Théories du module de satisfiabilité.

En ce qui concerne les analyseurs d'accès externe, l'analyseur d'accès IAM n'examine pas les journaux d'accès pour déterminer si une entité externe a effectivement accédé à une ressource dans votre zone de confiance. Par contre, il génère un résultat lorsqu'une politique basée sur les ressources autorise l'accès à une ressource, indépendamment du fait que la ressource ait été accédée par l'entité externe ou non.

En outre, l'analyseur d'accès IAM ne tient pas compte de l'état des comptes externes lorsqu'il détermine la situation. S'il indique que le compte 111122223333 peut accéder à votre compartiment Amazon S3, il ne dispose d'aucune information sur les utilisateurs, les rôles, les politiques de contrôle des services (SCP) ou d'autres configurations pertinentes de ce compte. Il s'agit d'une mesure de confidentialité, car l'analyseur d'accès IAM ne sait pas qui est propriétaire de l'autre compte. C'est également une question de sécurité, car il est important de connaître l'existence d'un accès externe potentiel, même si aucun principal actif n'est actuellement en mesure de l'utiliser.

L'analyseur d'accès IAM ne prend en compte que certaines clés de condition IAM que les utilisateurs externes ne peuvent pas influencer directement ou qui ont un autre impact sur l'autorisation. Pour obtenir des exemples de clés que l'IAM Access Analyzer prend en compte veuillez consulter IAM Access Analyzer filter keys (Clés de filtre Access Analyzer).

IAM Access Analyzer ne publie actuellement aucun résultat provenant des Service AWS principaux ou des comptes de service internes. Dans les rares cas où il n'est pas en mesure de déterminer avec certitude si une instruction accorde l'accès à une entité externe, il s'abstient de déclarer un faux positif. En effet, l'analyseur d'accès IAM est conçu pour fournir une vue exhaustive du partage des ressources dans votre compte et pour minimiser les faux négatifs.

Résultats relatifs à l'accès interne

Pour utiliser l'analyse d'accès interne, vous devez d'abord configurer l'analyseur en sélectionnant les ressources spécifiques que vous souhaitez surveiller. Une fois configurés, les résultats d'accès interne sont générés lorsqu'un principal (utilisateur ou rôle IAM) au sein de votre organisation ou de votre compte a accès aux ressources que vous avez sélectionnées. Un nouveau résultat est généré la prochaine fois que l'analyseur analyse les ressources spécifiées et identifie un principal ayant accès à ces ressources. Si une politique mise à jour autorise un principal déjà identifié dans un résultat, mais avec des autorisations ou des conditions différentes, un nouveau résultat est généré pour cette instance du partage de ressource. Cette politique mise à jour peut être une politique basée

sur les ressources, une politique basée sur l'identité, une politique de contrôle des services (SCP) ou une politique de contrôle des ressources (RCP). L'analyseur d'accès IAM évalue également les configurations de contrôle d'accès établies par des politiques déclaratives.

> **Note**
>
> Les résultats d'accès internes ne sont disponibles qu'à l'aide de l'action de l'API ListFindings V2.

Processus de génération de résultats relatifs aux accès internes par l'analyseur d'accès IAM

Pour analyser les accès internes, vous devez créer un analyseur distinct pour les résultats d'accès internes pour vos ressources, même si vous avez déjà créé un analyseur pour générer des résultats des accès externes ou des résultats des accès non utilisés.

Après avoir créé l'analyseur d'accès interne, l'analyseur d'accès IAM évalue toutes les politiques basées sur les ressources, les politiques basées sur l'identité, les politiques de contrôle des services (SCP), les politiques de contrôle des ressources (RCP) et les limites des autorisations au sein du compte ou de l'organisation que vous avez spécifiés.

En créant un analyseur dédié à l'accès interne aux ressources sélectionnées, vous pouvez identifier :

- quand un principal de votre organisation ou de votre compte peut accéder aux ressources sélectionnées ;
- l'ensemble des autorisations effectives accordées à un principal en fonction de l'intersection de toutes les politiques applicables ;
- les chemins d'accès complexes où un principal obtient l'accès en fonction de la combinaison des politiques d'identité et des politiques de ressources.

> **Note**
>
> L'analyseur d'accès IAM ne peut pas générer de résultats d'accès internes pour les organisations qui contiennent plus de 70 000 principaux (utilisateurs et rôles IAM combinés).

Résultats des accès non utilisés

Les résultats des accès non utilisés sont générés pour les entités IAM (principaux) au sein du compte ou de l'organisation sélectionné en fonction du nombre de jours spécifié lors de la création de l'analyseur. Un nouveau résultat est généré la prochaine fois que l'analyseur analyse les entités si l'une des conditions suivantes est remplie :

- Un rôle est inactif pendant le nombre de jours spécifié.
- Une autorisation, un mot de passe utilisateur ou une clé d'accès utilisateur non utilisés dépasse le nombre de jours spécifié.

> **Note**
>
> Les résultats d'accès non utilisés ne sont disponibles qu'à l'aide de l'action de l'API ListFindings V2.

Processus de génération de résultats relatifs aux accès non utilisés par l'analyseur d'accès IAM

Pour analyser les accès non utilisés, vous devez créer un analyseur distinct pour les résultats d'accès non utilisés pour vos rôles, même si vous avez déjà créé un analyseur pour générer des résultats des accès externes ou internes pour vos ressources.

Après avoir créé l'analyseur d'accès externes, l'analyseur d'accès IAM examine l'activité d'accès pour identifier les accès non utilisés. IAM Access Analyzer examine les dernières informations consultées pour tous les utilisateurs IAM, les rôles IAM, y compris les rôles de service, les clés d'accès utilisateur et les mots de passe utilisateur au sein de votre organisation et de vos AWS comptes. Cela vous permet d'identifier les accès non utilisés.

> **Note**
>
> Un rôle lié à un service est un type spécial de rôle de service lié à un service Service AWS et détenu par celui-ci. Service-linked les rôles ne sont pas analysés par des analyseurs d'accès non utilisés.

Pour les rôles et utilisateurs IAM actifs, l'analyseur d'accès IAM utilise les informations du dernier accès aux services et aux actions IAM pour identifier les autorisations non utilisées. Cela vous permet d'adapter votre processus de révision au niveau de l' AWS organisation et du compte. Vous pouvez également utiliser les informations relatives au dernier accès à l'action pour approfondir l'analyse des rôles individuels. Cela fournit des informations plus détaillées sur les autorisations spécifiques qui ne sont pas utilisées.

En créant un analyseur dédié aux accès non utilisés, vous pouvez examiner et identifier de manière exhaustive les accès non utilisés dans votre AWS environnement, en complétant ainsi les résultats générés par votre analyseur d'accès externe existant.

Démarrage avec AWS Identity and Access Management Access Analyzer

Utilisez les informations de cette rubrique pour en savoir plus sur les exigences nécessaires pour utiliser et gérer AWS Identity and Access Management Access Analyzer.

Si vous utilisez IAM Access Analyzer au sein d'une AWS Organizations organisation, c'est possible. Ajout d'un administrateur délégué pour l'analyseur d'accès IAM Pour plus d'informations sur les limites de service, consultezQuotas de l'IAM Access Analyzer.

Autorisations requises pour l'utilisation de l'IAM Access Analyzer

Pour configurer et utiliser correctement l'IAM Access Analyzer, le compte que vous utilisez doit disposer des autorisations requises.

AWS politiques gérées pour IAM Access Analyzer

AWS Identity and Access Management Access Analyzer fournit des politiques AWS gérées pour vous aider à démarrer rapidement.

- IAMAccessAnalyzerFullAccess- Permet un accès complet à IAM Access Analyzer pour les administrateurs. Cette politique permet également de créer les rôles liés au service dont l'IAM Access Analyzer a besoin pour analyser les ressources de votre compte ou AWS organisation
- IAMAccessAnalyzerReadOnlyAccess- Permet un accès en lecture seule à IAM Access Analyzer. Vous devez ajouter des politiques supplémentaires à vos identités IAM (utilisateurs, groupes d'utilisateurs ou rôles) pour les autoriser à afficher leurs résultats.

Ressources définies par l'analyseur d'accès IAM

Pour afficher les ressources définies par l'analyseur d'accès IAM, consultez Types de ressources définis par l'analyseur d'accès IAM dans la Référence de l'autorisation de service.

Autorisations de service de l'IAM Access Analyzer requises

L'analyseur d'accès IAM utilise un rôle lié au service (SLR) IAM nommé `AWSServiceRoleForAccessAnalyzer`. Ce reflex accorde au service un accès en lecture seule pour analyser les ressources à l'aide de politiques basées sur les AWS ressources et analyser les accès non utilisés en votre nom. Le service crée le rôle dans votre compte dans les scénarios suivants :

- Vous créez un analyseur d'accès externe avec votre compte comme zone de confiance.
- Vous créez un analyseur d'accès non utilisé avec votre compte comme compte sélectionné.
- Vous créez un analyseur d'accès interne avec votre compte comme zone de confiance.

Pour de plus amples informations, veuillez consulter Utilisation de rôles liés à un service pour AWS Identity and Access Management Access Analyzer.

> **Note**
>
> L'IAM Access Analyzer est régional. Pour les accès externes et internes, vous devez activer l'analyseur d'accès IAM dans chaque région de manière indépendante.
> En cas d'accès non utilisé, les résultats de l'analyseur ne changent pas en fonction de la région. Il n'est pas nécessaire de créer un analyseur dans chaque région où vous disposez de ressources.

Dans certains cas, une fois que vous avez créé un analyseur dans l'analyseur d'accès IAM, la page ou le tableau de bord des Résultats se charge sans résultats ni récapitulatif. Cela peut être dû à un retard d'affichage de vos résultats dans la console. Vous devrez peut-être actualiser manuellement le navigateur ou y revenir ultérieurement pour afficher vos résultats ou votre récapitulatif. Si vous ne voyez toujours pas de résultats pour un analyseur d'accès externe, c'est parce que vous n'avez pas de ressources prises en charge dans votre compte auxquelles une entité externe peut accéder. Si une politique qui accorde l'accès à une entité externe est appliquée à une ressource, l'IAM Access Analyzer génère un résultat.

> **Note**
>
> Pour les analyseurs d'accès externe, il peut s'écouler jusqu'à 30 minutes après la modification d'une politique pour que l'analyseur d'accès IAM analyse la ressource, puis génère un nouveau résultat ou met à jour un résultat existant pour l'accès à la ressource. Lorsque vous créez un analyseur d'accès interne, plusieurs minutes ou plusieurs heures peuvent être nécessaires avant que les résultats soient disponibles. Après l'analyse initiale, l'analyseur d'accès IAM réanalyse automatiquement toutes les politiques toutes les 24 heures.
> Pour tous les types d'analyseurs d'accès, les mises à jour des résultats peuvent ne pas apparaître immédiatement dans le tableau de bord.

Autorisations de l'analyseur d'accès IAM requises pour consulter le tableau de bord des résultats

Pour afficher le Tableau de bord des résultats de l'analyseur d'accès IAM, le compte que vous utilisez doit disposer d'un accès pour effectuer les actions requises suivantes :

- `GetAnalyzer`
- `ListAnalyzers`
- `GetFindingsStatistics`

Pour afficher toutes les actions définies par l'analyseur d'accès IAM, consultez Actions définies par l'analyseur d'accès IAM dans la Référence de l'autorisation de service.

Statut de l'IAM Access Analyzer

Pour afficher l'état de vos analyseurs, sélectionnez Analyzers (Analyseurs). Les analyseurs créés pour une organisation ou un compte peuvent avoir l'état suivant :

Statut	Description
Actif	Pour les analyseurs d'accès externe et externe, l'analyseur surveille activement les ressources dans sa zone de confiance. L'analyseur génère activement de nouveaux résultats et met à jour les résultats existants.

Statut	Description
	Pour les analyseurs d'accès non utilisés, l'analyseur surveille activement les accès non utilisés au sein de l'organisation sélectionnée ou pendant la Compte AWS période de suivi spécifiée. L'analyseur génère activement de nouveaux résultats et met à jour les résultats existants.
Création	La création de l'analyseur est toujours en cours. L'analyseur deviendra actif une fois la création terminée.
Désactivé	L'analyseur est désactivé suite à une action de l' AWS Organizations administrateur. Par exemple, le compte de l'analyseur en tant qu'administrateur délégué pour IAM Access Analyzer a pu être supprimé. Lorsque l’analyseur est désactivé, il ne génère pas de nouveaux résultats ou ne met pas à jour les résultats existants.
Échec	La création de l'analyseur a échoué en raison d'un problème de configuration. L'analyseur ne génère pas de résultats. Supprimez l'analyseur et créez-en un nouveau.

Création d’un analyseur d’accès IAM pour l’analyse de l’accès externe

Pour activer un analyseur d’accès externe dans une région, vous devez créer un analyseur dans cette région. Vous devez créer un analyseur d’accès externe dans chaque région dans laquelle vous souhaitez surveiller l’accès à vos ressources.

> **Note**
>
> Après avoir créé ou mis à jour un analyseur, il peut s'écouler un certain temps avant que les résultats soient disponibles.

Créez un analyseur d'accès externe avec Compte AWS la zone de confiance

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Sous Analyseur d'accès, sélectionnez Paramètres de l'analyseur.
3. Sélectionnez Create Analyzer (créer un analyseur).
4. Dans la section Analyse, sélectionnez Analyse des ressources - Accès externe.
5. Dans la section Informations sur l'analyseur, vérifiez que la région affichée est celle dans laquelle vous souhaitez activer l'analyseur d'accès IAM.
6. Entrez un nom pour l'analyseur.
7. Choisissez Compte actuel comme zone de confiance pour l'analyseur.

   > **Note**
   >
   > Si votre compte n'est pas le compte AWS Organizations de gestion ou le compte administrateur délégué, vous ne pouvez créer qu'un seul analyseur avec votre compte comme zone de confiance.

8. Facultatif. Ajoutez les balises que vous souhaitez appliquer à l'analyseur.
9. Sélectionnez Create Analyzer (créer un analyseur).

Lorsque vous créez un analyseur d'accès externe pour activer l'analyseur d'accès IAM, un rôle lié au service nommé `AWSServiceRoleForAccessAnalyzer` est créé dans votre compte.

Création d'un analyseur d'accès externe avec l'organisation comme zone de confiance

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Sous Analyseur d'accès, sélectionnez Paramètres de l'analyseur.
3. Sélectionnez Create Analyzer (créer un analyseur).
4. Dans la section Analyse, sélectionnez Analyse des ressources - Accès externe.

5. Dans la section Informations sur l'analyseur, vérifiez que la région affichée est celle dans laquelle vous souhaitez activer l'analyseur d'accès IAM.
6. Entrez un nom pour l'analyseur.
7. Choisissez Organisation courante comme zone de confiance pour l'analyseur.
8. Facultatif. Ajoutez les balises que vous souhaitez appliquer à l'analyseur.
9. Sélectionnez Soumettre.

Lorsque vous créez un analyseur d'accès externe avec l'organisation comme zone de confiance, un rôle lié au service nommé `AWSServiceRoleForAccessAnalyzer` est créé dans chaque compte de votre organisation.

Gestion d'un analyseur d'accès IAM pour l'analyse de l'accès externe

Pour activer un analyseur d'accès externe dans une région, vous devez créer un analyseur dans cette région. Vous devez créer un analyseur d'accès externe dans chaque région dans laquelle vous souhaitez surveiller l'accès à vos ressources.

> **Note**
>
> Après avoir créé ou mis à jour un analyseur, il peut s'écouler un certain temps avant que les résultats soient disponibles.

Mise à jour d'un analyseur d'accès externe

Utilisez la procédure suivante pour mettre à jour un analyseur d'accès externe.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Sous Analyseur d'accès, sélectionnez Paramètres de l'analyseur.
3. Dans la section Analyseurs, choisissez le nom de l'analyseur d'accès externe à gérer.
4. Dans l'onglet Règles d'archivage, vous pouvez créer, modifier ou supprimer des règles d'archivage pour l'analyseur. Pour de plus amples informations, veuillez consulter Règles d'archivage.
5. Dans l'onglet Balises, vous pouvez gérer et créer des balises pour l'analyseur. Pour de plus amples informations, veuillez consulter Tags pour Gestion des identités et des accès AWS resources.

Suppression d'un analyseur d'accès externe

Utilisez la procédure suivante pour supprimer un analyseur d'accès externe. Lorsque vous supprimez un analyseur, les ressources ne sont plus surveillées et aucun nouveau résultat n'est généré. Tous les résultats générés par l'analyseur sont supprimés.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Sous Analyseur d'accès, sélectionnez Paramètres de l'analyseur.
3. Dans la section Analyseurs, choisissez le nom de l'analyseur d'accès externe à supprimer.
4. Choisissez Supprimer l'analyseur.
5. Saisissez supprimer et choisissez Supprimer pour confirmer la suppression de l'analyseur.

Création d'un analyseur d'accès IAM pour l'analyse de l'accès interne

Pour activer un analyseur d'accès interne dans une région, vous devez créer un analyseur dans cette dernière. Vous devez créer un analyseur d'accès interne dans chaque région dans laquelle vous souhaitez surveiller l'accès à vos ressources.

L'analyseur d'accès IAM facture l'analyse des accès internes en fonction du nombre de ressources surveillées par analyseur et par mois. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

> **Note**
>
> Après avoir créé ou mis à jour un analyseur, il peut s'écouler un certain temps avant que les résultats soient disponibles.
> L'analyseur d'accès IAM ne peut pas générer de résultats d'accès internes pour les organisations qui contiennent plus de 70 000 principaux (utilisateurs et rôles IAM combinés).
> Vous ne pouvez créer qu'un seul analyseur d'accès interne au niveau de l'organisation dans une organisation AWS .

Créez un analyseur d'accès interne avec Compte AWS la zone de confiance

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Sous Analyseur d'accès, sélectionnez Paramètres de l'analyseur.
3. Sélectionnez Create Analyzer (créer un analyseur).

4. Dans la section Analyse, sélectionnez Analyse des ressources - Accès interne.
5. Dans la section Informations sur l'analyseur, vérifiez que la région affichée est celle dans laquelle vous souhaitez activer l'analyseur d'accès IAM.
6. Entrez un nom pour l'analyseur.
7. Choisissez Compte actuel comme zone de confiance pour l'analyseur.

> **Note**
>
> Si votre compte n'est pas le compte AWS Organizations de gestion ou le compte administrateur délégué, vous ne pouvez créer qu'un seul analyseur avec votre compte comme zone de confiance.

8. Dans la section Ressources à analyser, ajoutez les ressources à surveiller par l'analyseur.

   - Pour ajouter des ressources par compte, choisissez Ajouter > Ajouter des ressources à partir des comptes sélectionnés.

     1. Sélectionnez Tous les types de ressources pris en charge ou Définir des types de ressources spécifiques, puis sélectionnez les types de ressources dans la liste Type de ressource.

        Les analyseurs d'accès internes prennent en charge les types de ressources suivants :

        - Compartiments Amazon Simple Storage Service
        - Compartiments du répertoire Amazon Simple Storage Service
        - Instantanés de base de données service de base de données relationnelle Amazon
        - Instantanés de cluster de base de données service base de données relationnelle Amazon
        - Amazon DynamoDB Streams
        - Tables Amazon DynamoDB

     2. Choisissez Ajouter des ressources.

   - Pour ajouter des ressources par Amazon Resource Name (ARN), sélectionnez Ajouter des ressources > Ajouter des ressources en collant l'ARN de la ressource.

> **Note**
>
> ARNs doivent correspondre exactement ; les caractères génériques ne sont pas pris en charge. Pour Amazon S3, seuls les compartiments ARNs sont pris en charge. L'objet ARNs et les préfixes Amazon S3 ne sont pas pris en charge.

1. Pour chaque ARN de ressource, saisissez l'ID du propriétaire du compte et l'ARN de la ressource séparés par une virgule. Saisissez un ID de propriétaire du compte et un ARN de ressource par ligne.
2. Choisissez Ajouter des ressources.

- Pour ajouter des ressources à partir d'un fichier CSV, sélectionnez Ajouter des ressources > Ajouter des ressources en chargeant un fichier CSV.

  Vous pouvez utiliser Explorateur de ressources AWS pour rechercher des ressources dans vos comptes et exporter un fichier CSV. Vous pouvez ensuite charger le fichier CSV pour configurer les ressources que l'analyseur doit surveiller.

  1. Choisissez Choisir un fichier et sélectionnez le fichier CSV sur votre ordinateur.
  2. Choisissez Ajouter des ressources.

9. Facultatif. Ajoutez les balises que vous souhaitez appliquer à l'analyseur.
10. Sélectionnez Create Analyzer (créer un analyseur).

Lorsque vous créez un analyseur d'accès interne pour activer l'analyseur d'accès IAM, un rôle lié au service nommé `AWSServiceRoleForAccessAnalyzer` est créé dans votre compte.

Création d'un analyseur d'accès interne avec l'organisation comme zone de confiance

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Sous Analyseur d'accès, sélectionnez Paramètres de l'analyseur.
3. Sélectionnez Create Analyzer (créer un analyseur).
4. Dans la section Analyse, sélectionnez Analyse des ressources - Accès interne.
5. Dans la section Informations sur l'analyseur, vérifiez que la région affichée est celle dans laquelle vous souhaitez activer l'analyseur d'accès IAM.
6. Entrez un nom pour l'analyseur.

7. Choisissez Toute l'organisation comme zone de confiance pour l'analyseur.
8. Dans la section Ressources à analyser, ajoutez les ressources à surveiller par l'analyseur.

   - Pour ajouter des ressources pour le compte, choisissez Ajouter des ressources > Ajouter des ressources à partir des comptes sélectionnés.

     1. Sélectionnez Tous les types de ressources pris en charge ou Définir des types de ressources spécifiques, puis sélectionnez les types de ressources dans la liste Type de ressource.

        Les analyseurs d'accès internes prennent en charge les types de ressources suivants :

        - Compartiments Amazon Simple Storage Service
        - Compartiments du répertoire Amazon Simple Storage Service
        - Instantanés de base de données service de base de données relationnelle Amazon
        - Instantanés de cluster de base de données service base de données relationnelle Amazon
        - Amazon DynamoDB Streams
        - Tables Amazon DynamoDB

     2. Pour sélectionner des comptes au sein de votre organisation, choisissez Sélectionner dans l'organisation. Dans la section Sélectionner les comptes, choisissez Hiérarchie pour sélectionner les comptes en fonction de la structure organisationnelle ou Liste pour sélectionner les comptes à partir d'une liste de tous les comptes de votre organisation.

        Pour saisir manuellement les comptes de votre organisation, choisissez Enter AWS account ID. Entrez un ou plusieurs éléments Compte AWS IDs séparés par des virgules dans le champ ID du AWS compte.

     3. Choisissez Ajouter des ressources.

   - Pour ajouter des ressources par Amazon Resource Name (ARN), sélectionnez Ajouter des ressources > Ajouter des ressources en collant l'ARN de la ressource.

     > **Note**
     >
     > ARNs doivent correspondre exactement ; les caractères génériques ne sont pas pris en charge. Pour Amazon S3, seuls les compartiments ARNs sont pris en charge. L'objet ARNs et les préfixes Amazon S3 ne sont pas pris en charge.

1. Pour chaque ARN de ressource, saisissez l'ID du propriétaire du compte et l'ARN de la ressource séparés par une virgule. Saisissez un ID de propriétaire du compte et un ARN de ressource par ligne.
2. Choisissez Ajouter des ressources.

- Pour ajouter des ressources à partir d'un fichier CSV, sélectionnez Ajouter des ressources > Ajouter des ressources en chargeant un fichier CSV.

  Vous pouvez utiliser Explorateur de ressources AWS pour rechercher des ressources dans vos comptes et exporter un fichier CSV. Vous pouvez ensuite charger le fichier CSV pour configurer les ressources que l'analyseur doit surveiller.

  1. Choisissez Choisir un fichier et sélectionnez le fichier CSV sur votre ordinateur.
  2. Choisissez Ajouter des ressources.

9. Facultatif. Ajoutez les balises que vous souhaitez appliquer à l'analyseur.
10. Sélectionnez Soumettre.

Lorsque vous créez un analyseur d'accès interne avec l'organisation comme zone de confiance, un rôle lié au service nommé `AWSServiceRoleForAccessAnalyzer` est créé dans chaque compte de votre organisation.

Gestion d'un analyseur d'accès IAM pour l'analyse de l'accès interne

Pour activer un analyseur d'accès interne dans une région, vous devez créer un analyseur dans cette dernière. Vous devez créer un analyseur d'accès interne dans chaque région dans laquelle vous souhaitez surveiller l'accès à vos ressources.

> **Note**
>
> Après avoir créé ou mis à jour un analyseur, il peut s'écouler un certain temps avant que les résultats soient disponibles.

Mise à jour d'un analyseur d'accès interne

Utilisez la procédure suivante pour mettre à jour un analyseur d'accès interne.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.

2. Sous Analyseur d'accès, sélectionnez Paramètres de l'analyseur.
3. Dans la section Analyseurs, choisissez le nom de l'analyseur d'accès interne à gérer.
4. Dans l'onglet Règles d'archivage, vous pouvez créer, modifier ou supprimer des règles d'archivage pour l'analyseur. Pour de plus amples informations, veuillez consulter Règles d'archivage.
5. Dans l'onglet Balises, vous pouvez gérer et créer des balises pour l'analyseur. Pour de plus amples informations, veuillez consulter Tags pour Gestion des identités et des accès AWS resources.
6. Dans l'onglet Ressources, choisissez Modifier dans la section Ressources à analyser.

    a. Pour ajouter des ressources par compte, choisissez Ajouter des ressources > Ajouter des ressources à partir des comptes sélectionnés.

        1. Sélectionnez Tous les types de ressources pris en charge ou Définir des types de ressources spécifiques, puis sélectionnez les types de ressources dans la liste Type de ressource.

            Les analyseurs d'accès internes prennent en charge les types de ressources suivants :

            - Compartiments Amazon Simple Storage Service
            - Compartiments du répertoire Amazon Simple Storage Service
            - Instantanés de base de données service de base de données relationnelle Amazon
            - Instantanés de cluster de base de données service base de données relationnelle Amazon
            - Amazon DynamoDB Streams
            - Tables Amazon DynamoDB

        2. Choisissez Ajouter des ressources.

    b. Pour ajouter des ressources par Amazon Resource Name (ARN), sélectionnez Ajouter des ressources > Ajouter des ressources en collant l'ARN de la ressource.

        > **Note**
        >
        > ARNs doivent correspondre exactement ; les caractères génériques ne sont pas pris en charge. Pour Amazon S3, seuls les compartiments ARNs sont pris en charge. L'objet ARNs et les préfixes Amazon S3 ne sont pas pris en charge.

1. Pour chaque ARN de ressource, saisissez l'ID du propriétaire du compte et l'ARN de la ressource séparés par une virgule. Saisissez un ID de propriétaire du compte et un ARN de ressource par ligne.
2. Choisissez Ajouter des ressources.

c. Pour ajouter des ressources à partir d'un fichier CSV, sélectionnez Ajouter des ressources > Ajouter des ressources en chargeant un fichier CSV.

Vous pouvez utiliser Explorateur de ressources AWS pour rechercher des ressources dans vos comptes et exporter un fichier CSV. Vous pouvez ensuite charger le fichier CSV pour configurer les ressources que l'analyseur doit surveiller.

1. Choisissez Choisir un fichier et sélectionnez le fichier CSV sur votre ordinateur.
2. Choisissez Ajouter des ressources.

d. Pour supprimer des ressources de l'analyseur, cochez la case en regard des ressources à supprimer, puis sélectionnez Supprimer.

e. Sélectionnez Enregistrer les modifications.

> **Note**
>
> Toute mise à jour de l'analyseur sera évaluée lors de la prochaine analyse automatique dans les 24 heures.

Suppression d'un analyseur d'accès interne

Utilisez la procédure suivante pour supprimer un analyseur d'accès interne. Lorsque vous supprimez un analyseur, les ressources ne sont plus surveillées et aucun nouveau résultat n'est généré. Tous les résultats générés par l'analyseur sont supprimés.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Sous Analyseur d'accès, sélectionnez Paramètres de l'analyseur.
3. Dans la section Analyseurs, choisissez le nom de l'analyseur d'accès interne à supprimer.
4. Choisissez Supprimer l'analyseur.
5. Saisissez supprimer et choisissez Supprimer pour confirmer la suppression de l'analyseur.

Création d'un analyseur d'accès IAM pour l'analyse de l'accès non utilisé

Création d'un analyseur d'accès non utilisé pour le compte actuel

Suivez la procédure suivante pour créer un analyseur d'accès non utilisé pour un seul Compte AWS. En cas d'accès non utilisé, les résultats de l'analyseur ne changent pas en fonction de la région. Il n'est pas nécessaire de créer un analyseur dans chaque région où vous disposez de ressources.

L'analyseur d'accès IAM facture l'analyse des accès non utilisés en fonction du nombre de rôles et d'utilisateurs IAM analysés par mois et par analyseur. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

> **Note**
>
> Après avoir créé ou mis à jour un analyseur, il peut s'écouler un certain temps avant que les résultats soient disponibles.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Sous Analyseur d'accès, sélectionnez Paramètres de l'analyseur.
3. Sélectionnez Create Analyzer (créer un analyseur).
4. Dans la section Analyse, sélectionnez Analyse principale - Accès non utilisé.
5. Entrez un nom pour l'analyseur.
6. Pour Période de suivi, saisissez le nombre de jours d'analyse. L'analyseur évaluera uniquement les autorisations des entités IAM du compte sélectionné qui ont existé pendant toute la période de suivi. Par exemple, si vous définissez une période de suivi de 90 jours, seules les autorisations datant d'au moins 90 jours seront analysées, et les résultats seront générés s'ils ne montrent aucune utilisation pendant cette période. Vous pouvez saisir une valeur comprise entre 1 et 365 jours.
7. Dans la section Informations sur l'analyseur, vérifiez que la région affichée est celle dans laquelle vous souhaitez activer l'analyseur d'accès IAM.
8. Pour Comptes sélectionnés, choisissez Compte actuel.

> **Note**
>
> Si votre compte n'est pas le compte AWS Organizations de gestion ou le compte administrateur délégué, vous ne pouvez créer qu'un seul analyseur avec votre compte comme compte sélectionné.

9. Facultatif. Dans la section Exclure les personnes et les rôles IAM à l'aide de balises, vous pouvez spécifier des paires clé-valeur pour les utilisateurs et les rôles IAM à exclure de l'analyse des accès non utilisés. Les résultats ne seront pas générés pour les utilisateurs IAM et les rôles exclus correspondant aux paires clé-valeur. Pour la clé de balise, saisissez une valeur comprise entre 1 et 128 caractères. Elle ne peut pas commencer par `aws:`. Pour la valeur, vous pouvez spécifier une valeur comprise entre 0 et 256 caractères. Si vous ne saisissez pas de valeur, la règle est appliquée à tous les principaux ayant la clé de balise spécifiée. Choisissez Ajouter une nouvelle exclusion pour ajouter des paires clé-valeur supplémentaires à exclure.
10. Facultatif. Ajoutez les balises que vous souhaitez appliquer à l'analyseur.
11. Sélectionnez Create Analyzer (créer un analyseur).

Lorsque vous créez un analyseur d'accès non utilisé pour activer l'analyseur d'accès IAM, un rôle lié au service nommé `AWSServiceRoleForAccessAnalyzer` est créé dans votre compte.

Création d'un analyseur d'accès non utilisé avec l'organisation actuelle

Utilisez la procédure suivante pour créer un analyseur d'accès inutilisé permettant à une organisation de passer Comptes AWS en revue de manière centralisée l'ensemble de l'organisation. Pour l'analyse des accès non utilisés, les résultats de l'analyseur ne changent pas en fonction de la région. Il n'est pas nécessaire de créer un analyseur dans chaque région où vous disposez de ressources.

L'analyseur d'accès IAM facture l'analyse des accès non utilisés en fonction du nombre de rôles et d'utilisateurs IAM analysés par mois et par analyseur. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

> **Note**
>
> Si le compte d'un membre est supprimé de l'organisation, l'analyseur d'accès non utilisé cessera de générer de nouveaux résultats et de mettre à jour les résultats existants pour

ce compte au bout de 24 heures. Les résultats associés au compte membre supprimé de l'organisation seront définitivement supprimés après 90 jours.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Sous Analyseur d'accès, sélectionnez Paramètres de l'analyseur.
3. Sélectionnez Create Analyzer (créer un analyseur).
4. Dans la section Analyse, sélectionnez Analyse principale - Accès non utilisé.
5. Entrez un nom pour l'analyseur.
6. Pour Période de suivi, saisissez le nombre de jours d'analyse. L'analyseur évaluera uniquement les autorisations des entités IAM au sein des comptes de l'organisation sélectionnée qui ont existé pendant toute la période de suivi. Par exemple, si vous définissez une période de suivi de 90 jours, seules les autorisations datant d'au moins 90 jours seront analysées, et les résultats seront générés s'ils ne montrent aucune utilisation pendant cette période. Vous pouvez saisir une valeur comprise entre 1 et 365 jours.
7. Dans la section Informations sur l'analyseur, vérifiez que la région affichée est celle dans laquelle vous souhaitez activer l'analyseur d'accès IAM.
8. Pour Comptes sélectionnés, choisissez Organisation actuelle.
9. Facultatif. Dans la section Exclure Comptes AWS de l'analyse, vous pouvez choisir Comptes AWS dans votre organisation d'exclure de l'analyse des accès non utilisés. Les résultats ne seront pas générés pour les comptes exclus.

    a. Pour spécifier le compte individuel IDs à exclure, choisissez Spécifier l' Compte AWS ID et entrez le compte IDs séparé par des virgules dans le champ Compte AWS ID. Choisissez Exclure. Les comptes sont ensuite répertoriés dans le tableau Comptes AWS à exclure.

    b. Pour choisir parmi une liste des comptes de votre organisation à exclure, sélectionnez Choisir parmi l'organisation.

        i. Vous pouvez rechercher des comptes par nom, e-mail et ID de compte dans le champ Exclusion de comptes de l'organisation.

        ii. Choisissez Hiérarchie pour afficher vos comptes par unité d'organisation ou choisissez Liste pour afficher la liste de tous les comptes de votre organisation.

        iii. Choisissez Exclure tous les comptes actuels pour exclure tous les comptes d'une unité d'organisation ou choisissez Exclure pour exclure des comptes un par un.

Les comptes sont ensuite répertoriés dans le tableau Comptes AWS à exclure.

> **Note**
>
> Le compte propriétaire de l'analyseur d'organisation ne peut pas être l'un des comptes exclus. Lorsque de nouveaux comptes sont ajoutés à votre organisation, ils ne sont pas exclus de l'analyse, même si vous avez précédemment exclu tous les comptes actuels d'une unité d'organisation. Pour plus d'informations sur l'exclusion de comptes après la création d'un analyseur d'accès non utilisé, consultez Gestion d'un analyseur d'accès IAM pour l'analyse de l'accès non utilisé.

10. Facultatif. Dans la section Exclure les personnes et les rôles IAM à l'aide de balises, vous pouvez spécifier des paires clé-valeur pour les utilisateurs et les rôles IAM à exclure de l'analyse des accès non utilisés. Les résultats ne seront pas générés pour les utilisateurs IAM et les rôles exclus correspondant aux paires clé-valeur. Pour la clé de balise, saisissez une valeur comprise entre 1 et 128 caractères. Elle ne peut pas commencer par `aws:`. Pour la valeur, vous pouvez spécifier une valeur comprise entre 0 et 256 caractères. Si vous ne saisissez pas de valeur, la règle est appliquée à tous les principaux ayant la clé de balise spécifiée. Choisissez Ajouter une nouvelle exclusion pour ajouter des paires clé-valeur supplémentaires à exclure.
11. Facultatif. Ajoutez les balises que vous souhaitez appliquer à l'analyseur.
12. Sélectionnez Create Analyzer (créer un analyseur).

Lorsque vous créez un analyseur d'accès non utilisé pour activer l'analyseur d'accès IAM, un rôle lié au service nommé `AWSServiceRoleForAccessAnalyzer` est créé dans votre compte.

Gestion d'un analyseur d'accès IAM pour l'analyse de l'accès non utilisé

Utilisez les informations de cette rubrique pour apprendre à mettre à jour ou supprimer un analyseur d'accès non utilisé.

> **Note**
>
> Après avoir créé ou mis à jour un analyseur, il peut s'écouler un certain temps avant que les résultats soient disponibles.

Mise à jour d'un analyseur d'accès non utilisé

Utilisez la procédure suivante pour mettre à jour un analyseur d'accès non utilisé.

L'analyseur d'accès IAM facture l'analyse des accès non utilisés en fonction du nombre de rôles et d'utilisateurs IAM analysés par mois et par analyseur. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Sous Analyseur d'accès, sélectionnez Paramètres de l'analyseur.
3. Dans la section Analyseurs, choisissez le nom de l'analyseur d'accès non utilisé à gérer.
4. Dans l'onglet Exclusion, si l'analyseur a été créé pour une organisation comme champ d'analyse, choisissez Gérer dans la section Comptes AWS exclus.

    a. Pour spécifier le compte individuel IDs à exclure, choisissez Spécifier l' Compte AWS ID et entrez le compte IDs séparé par des virgules dans le champ Compte AWS ID. Choisissez Exclure. Les comptes sont ensuite répertoriés dans le tableau Comptes AWS à exclure.

    b. Pour choisir parmi une liste des comptes de votre organisation à exclure, sélectionnez Choisir parmi l'organisation.

        i. Vous pouvez rechercher des comptes par nom, e-mail et ID de compte dans le champ Exclusion de comptes de l'organisation.

        ii. Choisissez Hiérarchie pour afficher vos comptes par unité d'organisation ou choisissez Liste pour afficher la liste de tous les comptes de votre organisation.

        iii. Choisissez Exclure tous les comptes actuels pour exclure tous les comptes d'une unité d'organisation ou choisissez Exclure pour exclure des comptes un par un.

        Les comptes sont ensuite répertoriés dans le tableau Comptes AWS à exclure.

    c. Pour supprimer les comptes à exclure, choisissez Supprimer à côté du compte dans le tableau Comptes AWS à exclure.

    d. Sélectionnez Enregistrer les modifications.

> **Note**
>
> - Le compte propriétaire de l'analyseur d'organisation ne peut pas être l'un des comptes exclus.

- Lorsque de nouveaux comptes sont ajoutés à votre organisation, ils ne sont pas exclus de l'analyse, même si vous avez précédemment exclu tous les comptes actuels d'une unité d'organisation.
- Après avoir mis à jour les exclusions d'un analyseur, la mise à jour de la liste des comptes exclus peut prendre jusqu'à deux jours.

5. Dans l'onglet Exclusion, choisissez Gérer dans la section Utilisateurs et rôles IAM exclus avec des balises.

   a. Vous pouvez spécifier des paires clé-valeur pour les utilisateurs et les rôles IAM à exclure de l'analyse des accès non utilisés. Pour la clé de balise, saisissez une valeur comprise entre 1 et 128 caractères. Elle ne peut pas commencer par `aws:`. Pour la valeur, vous pouvez spécifier une valeur comprise entre 0 et 256 caractères. Si vous ne saisissez pas de valeur, la règle est appliquée à tous les principaux ayant la clé de balise spécifiée.

   b. Choisissez Ajouter une nouvelle exclusion pour ajouter des paires clé-valeur supplémentaires à exclure.

   c. Pour supprimer les paires clé-valeur à exclure, choisissez Supprimer en regard de la paire clé-valeur.

   d. Sélectionnez Enregistrer les modifications.

6. Dans l'onglet Règles d'archivage, vous pouvez créer, modifier ou supprimer des règles d'archivage pour l'analyseur. Pour de plus amples informations, veuillez consulter Règles d'archivage.

7. Dans l'onglet Balises, vous pouvez gérer et créer des balises pour l'analyseur. Pour de plus amples informations, veuillez consulter Tags pour Gestion des identités et des accès AWS resources.

Suppression d'un analyseur d'accès non utilisé

Suivez la procédure suivante pour supprimer un analyseur d'accès non utilisé. Lorsque vous supprimez un analyseur, les ressources ne sont plus surveillées et aucun nouveau résultat n'est généré. Tous les résultats générés par l'analyseur sont supprimés.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Sous Analyseur d'accès, sélectionnez Accès non utilisé.
3. Sous Analyseur d'accès, sélectionnez Paramètres de l'analyseur.

4. Dans la section Analyseurs, choisissez le nom de l’analyseur d’accès non utilisé à supprimer.
5. Choisissez Supprimer l’analyseur.
6. Saisissez supprimer et choisissez Supprimer pour confirmer la suppression de l’analyseur.

Affichage du tableau de bord des résultats de l’analyseur d’accès IAM

AWS Identity and Access Management Access Analyzer organise les résultats d'accès externes, internes et non utilisés dans un tableau de bord récapitulatif visuel. Le tableau de bord vous permet de mieux comprendre l'utilisation efficace des autorisations à grande échelle et d'identifier les comptes et les AWS ressources qui nécessitent une attention particulière. Vous pouvez utiliser le tableau de bord pour consulter les résultats par AWS organisation, compte et type de recherche.

Pour les résultats relatifs à l’accès externe et interne :

- Le tableau de bord met en évidence la distinction entre les résultats relatifs à l’accès public, les résultats relatifs à l’accès externe et les résultats relatifs à l’accès interne.
- Le tableau de bord fournit une division des résultats par type de ressource.

Pour les résultats des accès non utilisés

- Le tableau de bord met en évidence Comptes AWS les résultats d'accès les plus inutilisés.
- Le tableau de bord fournit une division des résultats par type.

Une fois que vous avez créé un analyseur d’accès, quel qu’en soit le type, l’analyseur d’accès IAM ajoute automatiquement les nouveaux résultats au tableau de bord correspondant. Cela vous permet d’identifier et de hiérarchiser les zones présentant le plus de problèmes de sécurité.

Les tableaux de bord récapitulatifs vous offrent une vue d'ensemble des problèmes d'accès détectés par IAM Access Analyzer dans votre environnement. AWS Vous pouvez ensuite examiner les résultats un par un pour approfondir les recherches et prendre les mesures appropriées.

Affichage du tableau de bord récapitulatif pour les analyseurs d'accès externes et internes

> **Note**
>
> Après avoir créé ou mis à jour un analyseur, le tableau de bord récapitulatif peut mettre du temps à refléter les mises à jour des résultats.

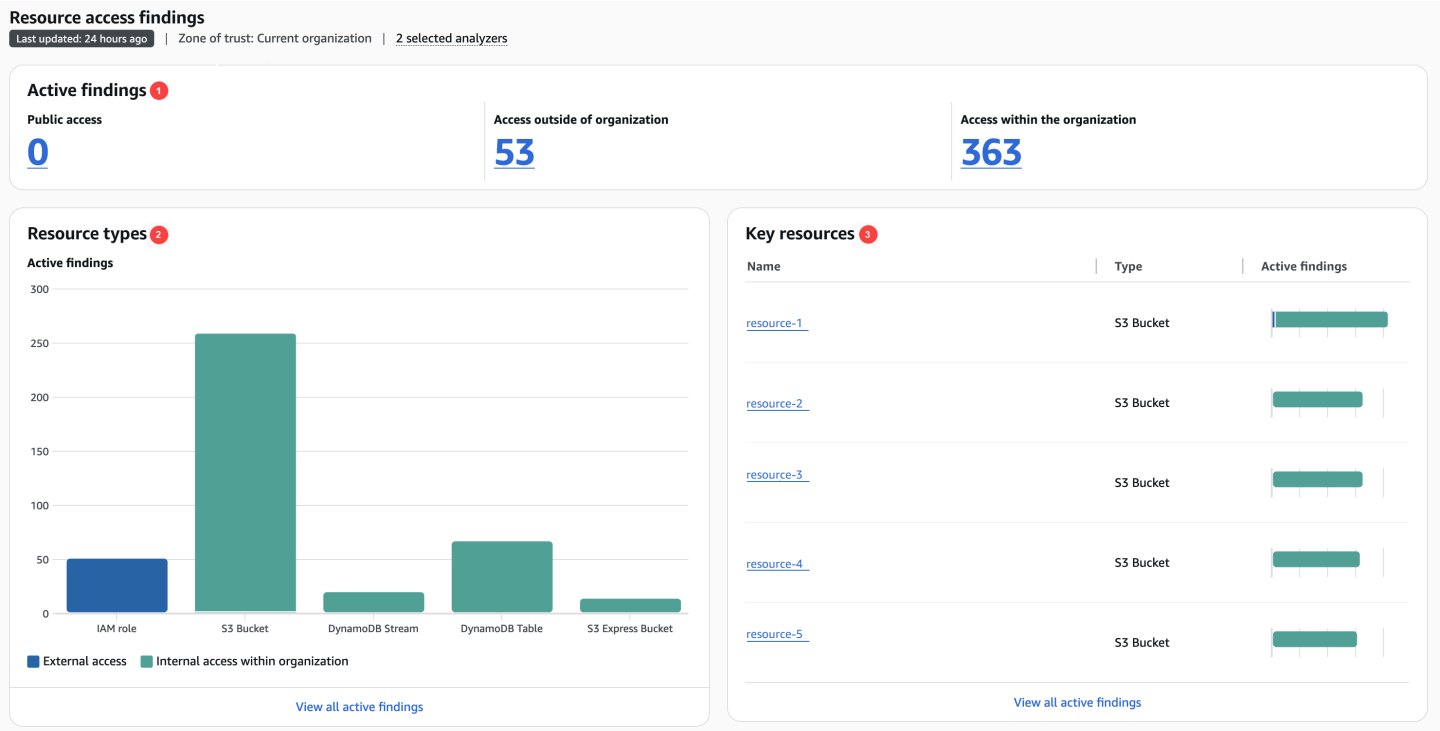
1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Analyseur d'accès. La fenêtre Récapitulatif s'affiche.
3. Choisissez Sélectionner les analyseurs.
4. Dans la fenêtre Sélectionner les analyseurs, choisissez Organisation ou Compte pour la Zone de confiance.

   > **Note**
   >
   > Seul le compte AWS Organizations de gestion ou l'administrateur délégué peut choisir Organisation comme zone de confiance.

5. Sélectionnez les analyseurs d'accès externes et internes dans le menu déroulant Analyseurs d'accès aux ressources.

   > **Note**
   >
   > Vous pouvez sélectionner au maximum un analyseur d'accès externe et un analyseur d'accès interne.

6. Choisissez Mettre à jour. Un résumé des résultats pour les analyseurs d'accès externes et internes sélectionnés est affiché dans la section Résultats de l'accès aux ressources.



Dans l'image précédente, le tableau de bord des résultats des accès externes est visible sur la page Récapitulatif :

1. La section Résultats actifs comprend le nombre de résultats actifs accessibles au public, le nombre de résultats actifs qui fournissent un accès en dehors du compte ou de l'organisation, et le nombre de résultats actifs accessibles en interne pour les analyseurs sélectionnés. Choisissez un nombre pour énumérer tous les résultats actifs de chaque type.
2. La section Types de ressources comprend une ventilation des types de ressources avec les résultats actifs pour les analyseurs sélectionnés. Sélectionnez Afficher tous les résultats actifs pour obtenir la liste complète des résultats actifs pour les analyseurs sélectionnés.
3. La section Ressources clés comprend un résumé des principales ressources contenant des résultats actifs. Ces informations vous aident à hiérarchiser les résultats pour vos ressources critiques. Sélectionnez Afficher tous les résultats actifs pour obtenir la liste complète des résultats actifs pour les analyseurs sélectionnés.

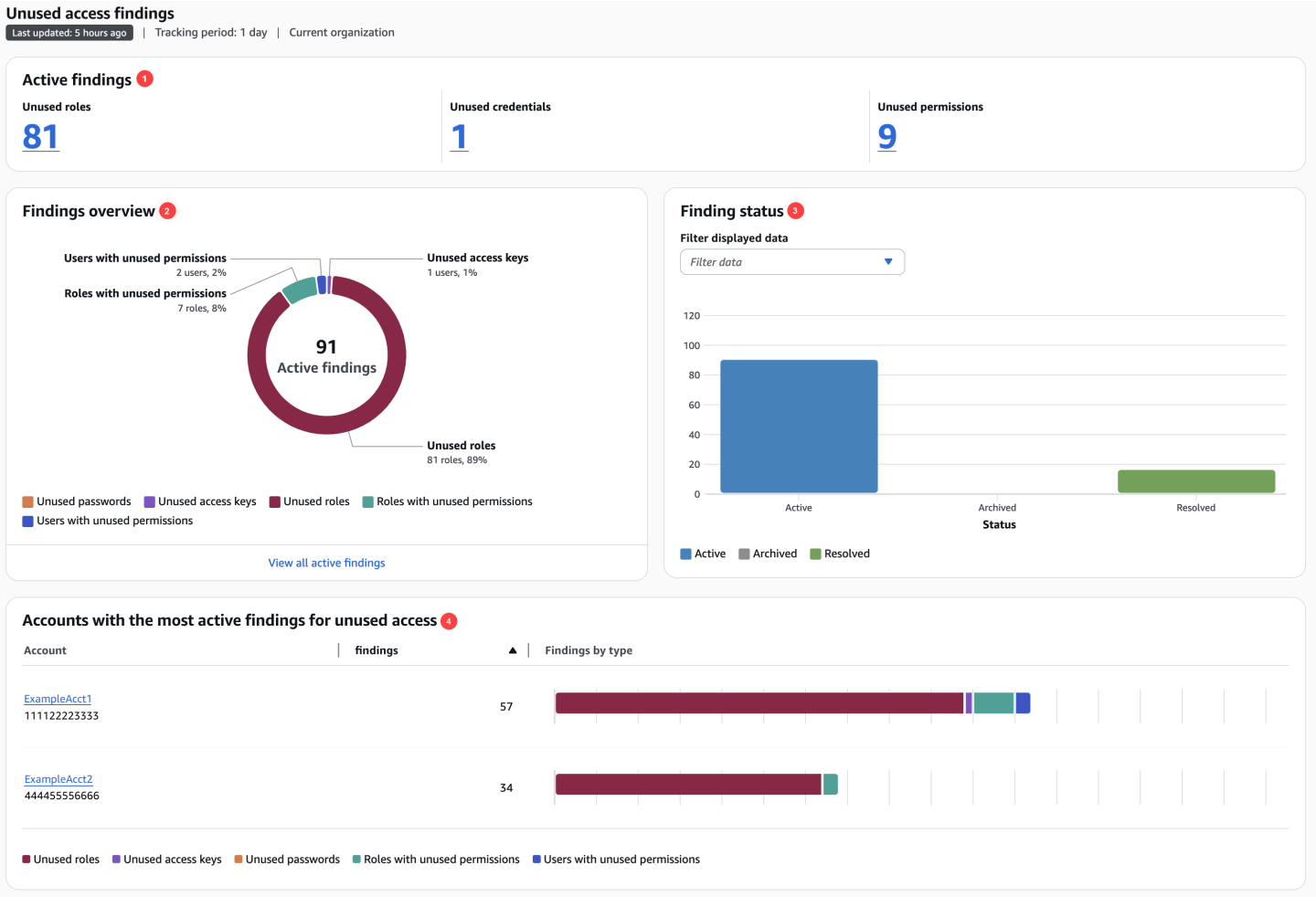
Affichage du tableau de bord récapitulatif des analyseurs d'accès non utilisés

> **Note**
>
> Après avoir créé ou mis à jour un analyseur, en fonction du nombre d'utilisateurs et de rôles, le tableau de bord récapitulatif peut mettre du temps à refléter les mises à jour des résultats.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Analyseur d'accès. La fenêtre Résumé de l'analyseur d'accès s'affiche.
3. Choisissez Sélectionner les analyseurs.
4. Dans la fenêtre Sélectionner les analyseurs, choisissez Organisation ou Compte pour la Zone de confiance.

   > **Note**
   >
   > Seul le compte AWS Organizations de gestion ou l'administrateur délégué peut choisir Organisation comme zone de confiance.

5. Choisissez un analyseur d'accès non utilisé dans le menu déroulant Analyseur d'accès non utilisé.
6. Choisissez Mettre à jour le résumé. Un résumé des résultats pour l'analyseur d'accès non utilisé sélectionné s'affiche dans la section Résultats d'accès non utilisés.



Dans l'image précédente, le tableau de bord des résultats des accès non utilisés est visible sur la page Récapitulatif :

1. La section Résultats actifs inclue le nombre de résultats actifs concernant des rôles, des informations d'identification et des autorisations non utilisés dans votre compte ou votre organisation. Les informations d'identification non utilisées incluent à la fois les clés d'accès et les résultats de mots de passe non utilisés. Les autorisations non utilisées incluent à la fois les utilisateurs et les rôles dont les autorisations ne sont pas utilisées. Choisissez un nombre pour énumérer tous les résultats actifs de chaque type.
2. La section Aperçu des résultats comprend une ventilation des types de résultats actifs. Choisissez Afficher tous les résultats actifs pour obtenir une liste complète des résultats actifs pour le compte ou l'organisation de l'analyseur.
3. La section Statut du résultat inclut une ventilation du statut des résultats (Actif, Archivé et Résolu) pour votre compte ou votre organisation. Vous pouvez sélectionner les statuts des résultats à afficher dans le menu déroulant Filtrer les données affichées.

4. La section Comptes présentant le plus grand nombre de résultats pour les accès non utilisés ne s'affiche que si les comptes sélectionnés dans votre analyseur d'accès non utilisés se situent au niveau de l'organisation. Il inclut une ventilation des comptes de votre organisation ayant enregistré les résultats les plus actifs. Cette liste n'est pas exhaustive de tous les comptes de votre organisation. Votre analyseur peut contenir des résultats actifs pour d'autres comptes non répertoriés dans cette section.

Examen des résultats de l'analyseur d'accès IAM

Après l'activation d'IAM Access Analyzer, l'étape suivante est l' examen de tous les résultats afin de déterminer si l'accès identifié dans le résultat est intentionnel ou non. Vous pouvez également examiner les résultats pour déterminer les résultats simillaires pour un accès prévu, puis créer une règle d'archivage pour automatiquement archiver ces résultats. Vous pouvez également examiner les résultats archivés et résolus.

Vous devez examiner tous les résultats de votre compte pour déterminer si l'accès externe, interne ou non utilisé est attendu et approuvé. Si l'accès identifié dans le résultat est attendu, vous pouvez archiver le résultat. Lorsque vous archivez un résultat, le statut devient Archivé et le résultat est supprimé de la liste des résultats actifs. Le résultat n'est pas supprimé. Vous pouvez consulter vos résultats archivés à tout moment. Passez en revue tous les résultats de votre compte jusqu'à ce qu'il ne reste plus aucun résultat actif. Une fois que vous êtes à zéro résultat, vous savez que tous les nouveaux résultats actifs générés proviennent d'une modification récente de votre environnement.

> **Tip**
>
> Si vous avez un grand nombre de résultats à examiner, utilisez l'API ou les commandes AWS CLI d'IAM Access Analyzer pour récupérer et gérer vos résultats par programmation.

Pour examiner les résultats actifs pour tous les types d'analyseurs d'accès

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Access Analyzer (Analyseur d'accès). Le tableau de bord des résultats s'affiche.
3. Choisissez Sélectionner les analyseurs.
4. Dans la fenêtre Sélectionner les analyseurs, choisissez au maximum un analyseur d'accès externe et un analyseur d'accès interne dans la liste déroulante Analyseurs d'accès aux

ressources. Choisissez un analyseur d'accès non utilisé dans le menu déroulant Analyseur d'accès non utilisé.

5. Choisissez Mettre à jour le résumé. Un résumé des résultats actifs pour les analyseurs d'accès sélectionnés est affiché sur le tableau de bord. Sélectionnez un type de résultat dans les sections Résultats d'accès aux ressources ou Résultats d'accès non utilisés pour afficher tous les résultats actifs du type sélectionné.

   Pour de plus d'informations sur l'affichage du tableau de bord des résultats, consultez Affichage du tableau de bord des résultats de l'analyseur d'accès IAM.

> **Note**
>
> Les résultats sont affichés uniquement si vous avez l'autorisation d'afficher les résultats de l'analyseur.

Résultats relatifs à l'accès externe et interne

> **Note**
>
> L'analyseur d'accès IAM facture l'analyse des accès internes en fonction du nombre de ressources surveillées par région et par mois. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

1. Sous Analyseur d'accès, sélectionnez Analyse des ressources.
2. Choisissez Sélectionner les analyseurs.
3. Dans la fenêtre Sélectionner les analyseurs, choisissez au maximum un analyseur d'accès externe et un analyseur d'accès interne dans la liste déroulante Analyseurs d'accès aux ressources.
4. Choisissez Mettre à jour le résumé.

   La page Analyse des ressources affiche les détails suivants concernant les ressources présentant des résultats actifs pour les analyseurs d'accès sélectionnés :

Nom

Nom de la ressource contenant des résultats actifs.

Type

Type de la ressource.

Compte propriétaire

Cette colonne s'affiche uniquement si vous utilisez une organisation comme zone de confiance pour un ou plusieurs analyseurs sélectionnés. Compte de l'organisation propriétaire de la ressource déclarée dans le résultat.

Résultats actifs

Représentation visuelle du nombre et du type de résultats actifs pour la ressource. Passez le curseur sur le champ pour afficher plus d'informations sur les résultats de la ressource.

Accès public

Indique si l'un des résultats de la ressource est autorise l'accès public.

Résultats des accès non utilisés

> **Note**
>
> L'analyseur d'accès IAM facture l'analyse d'accès non utilisé en fonction du nombre de rôles et d'utilisateurs IAM analysés par mois. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

1. Sous Analyseur d'accès, sélectionnez Accès non utilisé.
2. Choisissez Sélectionner les analyseurs.
3. Dans la fenêtre Sélectionner les analyseurs, choisissez un analyseur d'accès non utilisé dans la liste déroulante Analyseurs d'accès non utilisés.
4. Choisissez Mettre à jour le résumé.

   La page Accès non utilisés affiche les détails suivants concernant les entités IAM qui ont généré les résultats pour l'analyseur d'accès sélectionné :

ID de résultat

ID unique affecté au résultat. Choisissez l’ID du résultat pour afficher des détails supplémentaires sur l’entité IAM qui a généré le résultat.

Type de résultat

Type de résultat d’accès non utilisé : Clé d’accès non utilisée, Mot de passe non utilisé, Autorisation non utilisée, ou Rôle non utilisé.

Entité IAM

L’entité IAM mentionnée dans le résultat. Il peut s’agir d’un utilisateur ou d’un rôle IAM.

Compte AWS ID

Cette colonne s’affiche uniquement si vous configurez l’analyseur pour tous les Comptes AWS de l’organisation. Compte AWS Dans l'organisation propriétaire de l'entité IAM indiquée dans le résultat.

Dernière mise à jour

La dernière fois que l’entité IAM indiquée dans le résultat a été mise à jour, ou la date de création de l’entité si aucune mise à jour n’a été effectuée.

Statut

Statut du résultat : (Actif, Archivé ou Résolu).

Filtrage des résultats de l’analyseur d’accès IAM

Le filtrage par défaut de la page de résultat consiste à afficher tous les résultats actifs. Pour afficher tous les résultats, choisissez Tout dans le menu déroulant État. Pour afficher les résultats archivés, choisissez Archivé. Pour afficher les résultats résolus, choisissez Résolu. Lorsque vous commencez à utiliser IAM Access Analyzer, il n'y a aucun résultat archivé.

Utilisez des filtres pour afficher uniquement les résultats qui répondent aux critères de propriété spécifiés. Pour créer un filtre, sélectionnez la propriété à filtrer, puis choisissez si la propriété est égale ou contient une valeur, puis saisissez ou choisissez une valeur de propriété à filtrer.

Pour obtenir la liste des clés de filtre que vous pouvez utiliser pour créer ou mettre à jour une règle d'archivage, veuillez consulter Clés de filtre de l’IAM Access Analyzer.

Filtrage des ressources avec résultats actifs

Vous pouvez afficher et filtrer les résultats actifs par ressource pour un maximum d'un analyseur d'accès externe et un maximum d'un analyseur d'accès interne.

Pour filtrer les ressources avec des résultats actifs

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Analyse des ressources.
3. Pour filtrer par nom de ressource, saisissez tout ou partie du nom de la ressource dans le champ de recherche.
4. Dans le menu déroulant Filtrer le type d'accès, choisissez le type d'accès :
   - Tous les types : affiche les ressources avec tous les types de résultats d'accès.
   - Accès public : affiche uniquement les ressources dont les résultats sont accessibles au public.
   - Accès externe : affiche uniquement les ressources avec des résultats d'accès externe.
   - Accès interne au sein de l'organisation : affiche uniquement les ressources avec les résultats d'accès interne.
5. Dans le menu déroulant Filtrer le type de ressource, choisissez un type de ressource pour afficher uniquement les ressources du type sélectionné.

Filtrer les résultats des accès externes

Pour filtrer les résultats des accès externes

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Sélectionnez les paramètres de l'analyseur, puis choisissez l'analyseur d'accès externe dans la section Analyseurs.
3. Choisissez Afficher les résultats.
4. Choisissez la case de recherche pour afficher une liste des propriétés disponibles.
5. Choisissez la propriété à utiliser pour filtrer les résultats affichés.
6. Choisissez la valeur à faire correspondre pour la propriété. Seuls les résultats ayant cette valeur dans le résultat sont affichés.

   Par exemple, choisissez Ressource comme propriété, puis Ressource :, puis saisissez tout ou partie du nom d'un compartiment et appuyez sur Entrée. Seuls les résultats pour le compartiment

qui correspond aux critères du filtre sont affichés. Pour créer un filtre qui affiche uniquement les résultats pour les ressources autorisant l'accès public, vous pouvez choisir la propriété Accès public, puis Accès public =, puis Accès public = true.

Vous pouvez ajouter des propriétés supplémentaires pour filtrer davantage les résultats affichés. Lorsque vous ajoutez des propriétés supplémentaires, seuls les résultats correspondant à toutes les conditions du filtre sont affichés. La définition d'un filtre pour afficher les résultats correspondant à une propriété OU à une autre propriété n'est pas prise en charge. Choisissez Effacer les filtres pour effacer tous les filtres que vous avez définis et afficher tous les résultats avec le statut spécifié pour votre analyseur.

Certains champs ne s'affichent que lorsque vous veuillez consulter les résultats d'un analyseur dont la zone de confiance est une organisation.

Les propriétés suivantes sont disponibles pour définir des filtres pour l'accès externe :

- Public access (Accès public) : pour filtrer sur les résultats des ressources autorisant l'accès public, filtrez par Public access (Accès public), puis Public access: true (Accès public : vrai).
- Resource (Ressource) : pour filtrer par ressource, saisissez tout ou partie du nom de la ressource.
- Resource Type (Type de ressource) : pour filtrer par type de ressource, sélectionnez le type dans la liste affichée.
- Compte propriétaire de la ressource : utilisez cette propriété pour filtrer selon le compte de l'organisation propriétaire de la ressource indiquée dans le résultat.
- Restriction de la politique de contrôle des ressources : utilisez cette propriété pour filtrer selon le type de restriction appliqué par une politique de contrôle des ressources (RCP) Organizations. Pour en savoir plus, consultez la section Politiques de contrôle des ressources (RCPs) dans le guide de AWS Organizations l'utilisateur.
  - Échec de l'évaluation de la RCP : une erreur s'est produite lors de l'évaluation de la RCP.
  - Non applicable : aucun RCP ne restreint cette ressource ou ce principal. Cela inclut également les ressources qui ne RCPs sont pas encore prises en charge.
  - Applicable : l'administrateur de votre organisation a défini des restrictions via une RCP qui ont un impact sur la ressource ou le type de ressource. Contactez l'administrateur de votre organisation pour plus de détails.
- AWS Compte — Utilisez cette propriété pour filtrer en fonction Compte AWS des accès autorisés dans la section Principal d'une déclaration de politique. Pour filtrer par Compte AWS, saisissez tout

ou partie de l' Compte AWS identifiant à 12 chiffres, ou tout ou partie de l'ARN complet du compte de l' AWS utilisateur ou du rôle externe ayant accès aux ressources du compte courant.

- Utilisateur canonique : pour filtrer par utilisateur canonique, saisissez l'ID d'utilisateur canonique comme défini pour les compartiments Amazon S3. Pour en savoir plus, veuillez consulter Identificants de compte AWS.
- Federated User (Utilisateur fédéré) : pour filtrer par utilisateur fédéré, saisissez tout ou partie de l'ARN de l'identité fédérée. Pour en savoir plus, veuillez consulter Fournisseurs d'identité et fédération.
- ID de résultat : pour filtrer par ID de résultat, saisissez tout ou une partie de l'ID de résultat.
- Erreur : pour filtrer par type d'erreur, choisissez Accès refusé ou Erreur interne.
- ARN principal — Utilisez cette propriété pour filtrer l'ARN du principal (utilisateur, rôle ou groupe IAM) utilisé dans une clé de PrincipalArn condition aws :. Pour filtrer par ARN principal, saisissez tout ou partie de l'ARN de l'utilisateur, du rôle ou du groupe IAM provenant d'un utilisateur externe Compte AWS indiqué dans une recherche.
- OrGid principal — Pour filtrer par orGid principal, saisissez tout ou partie de l'ID d'organisation associé aux principaux externes appartenant à l'organisation spécifiée comme condition dans AWS le résultat. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- Principal OrgPaths — Pour filtrer par principal OrgPaths, saisissez tout ou partie de l'ID de l' AWS organisation ou de l'unité organisationnelle (UO) qui autorise l'accès à tous les principaux externes membres du compte de l'organisation ou de l'UO spécifiée comme condition dans la politique. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- Compte source : pour filtrer sur le compte source, saisissez tout ou partie de l' Compte AWS identifiant associé aux ressources, tel qu'il est utilisé dans certaines autorisations interservices dans AWS. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- Source ARN (ARN source) : pour filtrer par ARN source, saisissez tout ou partie de l'ARN spécifié comme condition dans le résultat. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- Source IP (IP source) : pour filtrer par IP source, saisissez tout ou partie de l'adresse IP qui permet aux entités externes d'accéder aux ressources du compte actuel lors de l'utilisation de l'adresse IP spécifiée. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- OrgID source : pour filtrer par OrgID source, saisissez tout ou une partie de l'ID d'organisation associé aux ressources, tel qu' utilisé dans certaines autorisations interservice dans AWS. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.

- Source OrgPaths : pour filtrer par source OrgPaths, saisissez tout ou partie de l'unité organisationnelle (UO) associée aux ressources, telle qu'elle est utilisée dans certaines autorisations interservices dans AWS. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- Source VPC (VPC source) : pour filtrer par VPC source, saisissez tout ou partie de l'ID de VPC qui permet aux entités externes d'accéder aux ressources du compte actuel lors de l'utilisation du VPC spécifié. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- ARN du VPC source : pour filtrer par ARN du VPC source, saisissez tout ou partie de l'ARN du VPC qui permet aux entités externes d'accéder aux ressources du compte courant lors de l'utilisation du VPC spécifié. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- VPCE source : pour filtrer par VPCE source, saisissez tout ou partie de l'ID du point de terminaison d'un VPC qui permet aux entités externes d'accéder aux ressources du compte actuel lors de l'utilisation du point de terminaison d'un VPC spécifié. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- Compte VPCE — Pour filtrer par compte VPCE, saisissez tout ou partie de l' Compte AWS identifiant à 12 chiffres qui possède les entités externes du point de terminaison VPC et permet aux entités externes d'accéder aux ressources. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- VPCE OrgID : pour filtrer par VPCE OrgID, saisissez tout ou partie de l'ID de l'organisation qui possède les entités externes du point de terminaison du VPC et autorise les entités externes à accéder aux ressources. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- VPCE OrgPaths — Pour filtrer par VPCE OrgPaths, saisissez tout ou partie de l'unité organisationnelle (UO) qui possède les entités externes du point de terminaison VPC et autorise les entités externes à accéder aux ressources. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- ID utilisateur — Pour filtrer par ID utilisateur, saisissez tout ou partie de l'ID utilisateur de l'utilisateur IAM auprès d'un utilisateur Compte AWS externe autorisé à accéder aux ressources du compte courant. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- ID de clé KMS : pour filtrer par ID de clé KMS, saisissez tout ou une partie de l'ID de la clé KMS spécifiée comme condition pour l'accès aux objets Amazon S3 chiffrés AWS KMS dans votre compte actuel.

- Mode session : pour filtrer par mode de session pour les compartiments de répertoires Amazon S3 (`ReadOnly` ou `ReadWrite` saisissez tout ou partie du mode session). Pour en savoir plus, consultez CreateSessionle manuel Amazon Simple Storage Service API Reference.
- Google Audience (Audience Google) : pour filtrer par audience Google, saisissez tout ou partie de l'ID de l'application Google spécifiée comme condition pour l'accès au rôle IAM dans votre compte actuel. Pour en savoir plus, veuillez consulter Clés de contexte de condition IAM et AWS STS.
- Audience Cognito : pour filtrer par audience Amazon Cognito, saisissez tout ou une partie de l'ID de réserve d'identités Amazon Cognito spécifié comme condition pour l'accès au rôle IAM dans votre compte actuel. Pour en savoir plus, veuillez consulter Clés de contexte de condition IAM et AWS STS.
- Compte appelant : Compte AWS ID du compte qui possède ou contient l'entité appelante, tel qu'un rôle IAM, un utilisateur ou un utilisateur root du compte. Ceci est utilisé par les services qui appellent AWS KMS. Pour filtrer par compte d'appelant, saisissez tout ou partie de l' Compte AWS identifiant.
- Facebook App ID (ID d'application Facebook) : pour filtrer par ID d'application Facebook, saisissez tout ou partie de l'ID d'application Facebook (ou de l'ID de site) spécifié comme condition pour autoriser la connexion avec accès de fédération Facebook à un rôle IAM dans votre compte actuel. Pour en savoir plus, consultez la section identifiant dans IAM et clés AWS STS contextuelles de condition.
- Amazon App ID (ID d'application Amazon) : pour filtrer par ID d'application Amazon, saisissez tout ou partie de l'ID d'application Amazon (ou de l'ID de site) spécifié comme condition pour autoriser l'accès de fédération Login with Amazon à un rôle IAM dans votre compte actuel. Pour en savoir plus, consultez la section identifiant dans IAM et clés AWS STS contextuelles de condition.
- Lambda Event Source Token (Jeton de source d'événement Lambda) : pour filtrer sur le jeton de source d'événement Lambda transmis avec des intégrations Alexa, saisissez tout ou partie de la chaîne du jeton.

Filtrer les résultats des accès internes

Pour filtrer les résultats des accès internes

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Sélectionnez les paramètres de l'analyseur, puis choisissez l'analyseur d'accès interne dans la section Analyseurs.
3. Choisissez Afficher les résultats.

4. Choisissez la case de recherche pour afficher une liste des propriétés disponibles.
5. Choisissez la propriété à utiliser pour filtrer les résultats affichés.
6. Choisissez la valeur à faire correspondre pour la propriété. Seuls les résultats ayant cette valeur dans le résultat sont affichés.

   Par exemple, choisissez Ressource comme propriété, puis Ressource :, puis saisissez tout ou partie du nom d'un compartiment et appuyez sur Entrée. Seuls les résultats pour le compartiment qui correspond aux critères du filtre sont affichés.

Vous pouvez ajouter des propriétés supplémentaires pour filtrer davantage les résultats affichés. Lorsque vous ajoutez des propriétés supplémentaires, seuls les résultats correspondant à toutes les conditions du filtre sont affichés. La définition d'un filtre pour afficher les résultats correspondant à une propriété OU à une autre propriété n'est pas prise en charge. Choisissez Effacer les filtres pour effacer tous les filtres que vous avez définis et afficher tous les résultats avec le statut spécifié pour votre analyseur.

Certains champs ne s'affichent que lorsque vous veuillez consulter les résultats d'un analyseur dont la zone de confiance est une organisation.

Les champs suivants ne s'affichent que lorsque vous consultez les résultats d'un analyseur qui surveille les accès internes :

- Resource (Ressource) : pour filtrer par ressource, saisissez tout ou partie du nom de la ressource.
- Resource Type (Type de ressource) : pour filtrer par type de ressource, sélectionnez le type dans la liste affichée.
- Compte propriétaire de la ressource : utilisez cette propriété pour filtrer selon le compte de l'organisation propriétaire de la ressource indiquée dans le résultat.
- ID de résultat : pour filtrer par ID de résultat, saisissez tout ou une partie de l'ID de résultat.

Filtrer les résultats des accès non utilisés

Pour filtrer les résultats des accès non utilisés

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Accès non utilisé, puis choisissez l'analyseur dans la menu déroulant Afficher l'analyseur.

3. Choisissez la case de recherche pour afficher une liste des propriétés disponibles.
4. Choisissez la propriété à utiliser pour filtrer les résultats affichés.
5. Choisissez la valeur à faire correspondre pour la propriété. Seuls les résultats ayant cette valeur dans le résultat sont affichés.

   Par exemple, choisissez Type de résultats comme propriété, choisissez ensuite Type de résultats =, puis Type de résultats = Rôle non utilisé. Seuls les résultats présentant le type Rôle non utilisé sont affichés.

Vous pouvez ajouter des propriétés supplémentaires pour filtrer davantage les résultats affichés. Lorsque vous ajoutez des propriétés supplémentaires, seuls les résultats correspondant à toutes les conditions du filtre sont affichés. La définition d'un filtre pour afficher les résultats correspondant à une propriété OU à une autre propriété n'est pas prise en charge. Choisissez Effacer les filtres pour effacer tous les filtres que vous avez définis et afficher tous les résultats avec le statut spécifié pour votre analyseur.

Les champs suivants ne s’affichent que lorsque vous consultez les résultats d’un analyseur qui surveille les accès non utilisés :

- Type de résultats : pour filtrer par type de résultat, filtrez par Type de résultats, puis choisissez le type de résultat.
- Resource (Ressource) : pour filtrer par ressource, saisissez tout ou partie du nom de la ressource.
- Resource Type (Type de ressource) : pour filtrer par type de ressource, sélectionnez le type dans la liste affichée.
- Compte propriétaire de la ressource : utilisez cette propriété pour filtrer selon le compte de l’organisation propriétaire de la ressource indiquée dans le résultat.
- ID de résultat : pour filtrer par ID de résultat, saisissez tout ou une partie de l’ID de résultat.

Archivage des résultats de l’analyseur d’accès IAM

Lorsque vous obtenez un résultat pour accéder à une ressource intentionnelle, vous pouvez archiver les résultats. Par exemple, un résultat des accès externes ou internes pour un compartiment Amazon S3 auquel on accède pour des flux de travail approuvés ou un résultat des accès non utilisés pour une clé d’accès qui peut toujours être nécessaire. Lorsque vous archivez un résultat, il est effacé de la liste des résultats actifs. Les résultats archivés ne sont pas supprimés. Vous pouvez filtrer la page Résultats pour afficher vos résultats archivés et les désarchiver à tout moment.

Pour archiver les résultats à partir de la page Findings (Résultats)

1. Sélectionnez la case à cocher en regard d’un ou de plusieurs résultats à archiver.
2. Choisissez Actions, puis Archiver.

   Une confirmation s'affiche en haut de l'écran.

Pour archiver les résultats à partir de la page Détails des résultats

1. Choisissez l'ID de résultat du résultat à archiver.
2. Choisissez Archive (Archiver).

   Une confirmation s'affiche en haut de l'écran.

Pour désarchiver les résultats, répétez les étapes précédentes, mais sélectionnez Unarchive (Désarchiver) plutôt que Archive (Archiver). Lorsque vous désarchivez un résultat, son statut devient Active (Actif).

Résolution des problèmes liés aux résultats de l’analyseur d’accès IAM

Résolution des problèmes liés aux résultats

Pour résoudre les problèmes liés aux résultats des accès externes et internes résultant d’un accès non intentionnel, modifiez l’instruction de politique pour supprimer les autorisations qui accordent l’accès à la ressource identifiée.

Pour les résultats liés aux compartiments S3, utilisez la console Amazon S3 pour configurer les autorisations sur le compartiment.

Pour les rôles IAM, utilisez la console IAM pour modifier la politique d'approbation du rôle IAM répertorié.

Pour les autres ressources prises en charge, utilisez la console pour modifier les instructions de politique ayant abouti à un résultat généré.

Lorsque vous apportez une modification afin de résoudre un problème de résultats de ressources, comme la modification d’une politique appliquée à un rôle IAM, l’analyseur d’accès IAM analyse la ressource à nouveau. Si l’accès à la ressource est supprimé, le statut du résultat devient Résolu. Le résultat sera ensuite affiché dans la liste des résultats résolus au lieu de la liste des résultats actifs.

> **Note**
>
> Cela ne s'applique pas aux résultats signalant une Erreur. Lorsque l'analyseur d'accès IAM n'est pas en mesure d'analyser une ressource, il génère un résultat signalant une erreur. Si vous résolvez le problème qui a empêché l'analyseur d'accès IAM d'analyser la ressource, le résultat d'erreur est complètement supprimé au lieu d'être transformé en résultat résolu. Pour de plus amples informations, veuillez consulter Résultats d'erreur de l'analyseur d'accès IAM.

Si les modifications que vous avez apportées ont entraîné un accès externe ou interne à la ressource, mais d'une manière différente, par exemple avec un principal différent ou pour une autorisation différente, l'analyseur d'accès IAM résoudra le résultat d'origine et générera un nouveau résultat actif. Si les modifications que vous avez apportées ont entraîné des erreurs internes ou des erreurs d'accès refusé, tous les résultats actifs sans erreur liés à l'accès spécifique à la ressource sont résolus et un nouveau résultat d'erreur est généré.

> **Note**
>
> Pour les analyseurs d'accès externes, lorsque la politique est modifiée, l'analyseur d'accès IAM peut prendre jusqu'à 30 minutes pour analyser à nouveau la ressource et mettre à jour le résultat.
> Pour les analyseurs d'accès internes, plusieurs minutes ou plusieurs heures peuvent être nécessaires à l'analyseur d'accès IAM pour analyser à nouveau la ressource, puis mettre à jour les résultats. L'analyseur d'accès IAM réanalyse automatiquement toutes les politiques toutes les 24 heures.
> Les résultats résolus sont supprimés 90 jours après la dernière mise à jour du résultat.

Résolution des problèmes liés aux résultats d'accès non utilisés

L'analyseur d'accès IAM propose des étapes recommandées pour résoudre les résultats non utilisés de l'analyseur d'accès en fonction du type de résultat.

Une fois que vous avez apporté une modification pour résoudre un résultat des accès non utilisés, le statut du résultat passe à Résolu lors de la prochaine exécution de l'analyseur d'accès non utilisé. Le résultat n'est plus affiché dans la liste des résultats actifs, mais plutôt dans la liste des résultats résolus. Si vous apportez une modification qui ne résoud que partiellement à un résultat des accès

non utilisés, le résultat existant devient Résolu mais un nouveau est généré. Par exemple, si vous ne supprimez que certaines autorisations non utilisées dans un résultat, mais pas toutes.

L'analyseur d'accès IAM facture l'analyse d'accès non utilisé en fonction du nombre de rôles et d'utilisateurs IAM analysés par mois. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

Résolution des problèmes liés aux résultats des autorisations non utilisées

Pour les résultats des autorisations non utilisées, l'analyseur d'accès IAM peut recommander des politiques à supprimer d'un rôle ou d'un utilisateur IAM et fournir de nouvelles politiques pour remplacer les politiques d'autorisation existantes. La recommandation de politique n'est pas prise en charge pour les scénarios suivants :

- Le résultat d'autorisation non utilisée concerne un utilisateur IAM appartenant à un groupe d'utilisateurs.
- Le résultat d'autorisation non utilisée concerne un rôle IAM pour IAM Identity Center.
- Le résultat d'autorisation non utilisée est associé à une politique d'autorisation existante qui inclut l'élément `notAction`.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Accès non utilisé.
3. Choisissez un résultat dont le Type de résultat est Autorisations non utilisées.
4. Dans la section Recommandations, si des politiques sont répertoriées dans la colonne Politique recommandée, choisissez Prévisualiser la politique pour afficher la politique existante avec la politique recommandée pour remplacer la politique existante. S'il existe plusieurs politiques recommandées, vous pouvez choisir la stratégie suivante et la stratégie précédente pour afficher chaque politique existante et recommandée.
5. Choisissez Télécharger le JSON pour télécharger un fichier .zip contenant les fichiers JSON de toutes les politiques recommandées.
6. Créez les politiques recommandées et attachez-les au rôle ou à l'utilisateur IAM. Pour plus d'informations, consultez Modification des autorisations d'un utilisateur (console) et Modification d'une politique d'autorisations d'un rôle (console).
7. Supprimez les politiques répertoriées dans la colonne Politique d'autorisation existante du rôle ou de l'utilisateur IAM. Pour plus d'informations, consultez Supprimer les autorisations d'un utilisateur (console) et Modifier la politique d'autorisations d'un rôle (console).

Résolution des problèmes liés aux rôles non utilisés

Pour les résultats de rôle non utilisé, l'analyseur d'accès IAM recommande de supprimer le rôle IAM non utilisé.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Accès non utilisé.
3. Choisissez un résultat dont le Type de résultat est Rôle non utilisé.
4. Dans la section Recommandations, passez en revue les détails du rôle IAM.
5. Supprimez le rôle IAM. Pour plus d'informations sur la suppression d'un rôle IAM, consultez Suppression d'un rôle IAM (console).

Résolution des problèmes liés aux résultats de clé d'accès non utilisée

Pour les résultats de clé d'accès non utilisée, l'analyseur d'accès IAM recommande de désactiver ou de supprimer la clé d'accès non utilisée.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Accès non utilisé.
3. Choisissez une recherche dont le Type de résultat est Clés d'accès non utilisées.
4. Dans la section Recommandations, passez en revue les détails de la clé d'accès.
5. Désactivez ou supprimez les clés d'accès. Pour de plus amples informations, consultez Gestion des clés d'accès (console).

Résolution des problèmes liés aux mots de passe non utilisés

Pour les résultats de mot de passe non utilisé, l'analyseur d'accès IAM recommande de supprimer le mot de passe non utilisé pour l'utilisateur IAM.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Accès non utilisé.
3. Choisissez une recherche dont le Type de résultat est Mot de passe non utilisé.
4. Dans la section Recommandations, passez en revue les informations relatives à l'utilisateur IAM.
5. Supprimez le mot de passe de l'utilisateur IAM. Pour plus d'informations, consultez Création, modification ou suppression d'un mot de passe utilisateur IAM (console).

Résultats d'erreur de l'analyseur d'accès IAM

Lorsque l'analyseur d'accès IAM analyse les ressources, il génère généralement des résultats qui indiquent qui a accès à vos ressources. Cependant, dans certains cas, l'analyseur peut rencontrer des problèmes qui l'empêchent de terminer l'analyse. En pareil cas, l'analyseur d'accès IAM génère des résultats d'erreur à la place.

Les résultats d'erreur indiquent que l'analyseur d'accès IAM n'a pas pu terminer l'analyse pour une ressource spécifique ou pour une paire principal-ressource spécifique. Ces résultats vous aident à identifier les ressources qui pourraient nécessiter une attention particulière afin de garantir une analyse correcte.

Résultats d'erreur des accès externes

Les analyseurs d'accès externes, qui identifient les ressources partagées en dehors de votre compte ou de votre organisation, peuvent générer deux types de résultats d'erreur :

- INTERNAL_ERROR : indique que l'analyseur d'accès IAM a rencontré un problème interne lors de l'analyse de la ressource. Cela peut être dû à des limites de service ou à des problèmes temporaires.


```
{
 "findingDetails": [
  {
   "externalAccessDetails": {}
  }
 ],
 "resource": "arn:aws:iam::941407043048:role/TestAccessAnalyzer",
 "status": "ACTIVE",
 "error": "INTERNAL_ERROR",
 "createdAt": "2022-07-14T01:31:43.085000+00:00",
 "resourceType": "AWS::IAM::Role",
 "findingType": "ExternalAccess",
 "resourceOwnerAccount": "941407043048",
 "analyzedAt": "2025-03-19T06:51:46.109000+00:00",
 "id": "4b035c7d-b7d2-40e4-a6c3-9887d1a995df",
 "updatedAt": "2022-07-14T01:31:43.085000+00:00"
}
```

- ACCESS_DENIED : indique que l'analyseur d'accès IAM ne dispose pas des autorisations requises pour analyser la ressource. Cela se produit généralement lorsque le rôle lié au service (SLR) pour l'analyseur d'accès IAM se voit refuser l'accès à la ressource.


```
{
 "findingDetails": [
  {
   "externalAccessDetails": {}
  }
 ],
 "resource": "arn:aws:kms:us-west-2:941407043048:key/01cae123-
b7f2-4488-9a05-0070a072ea2c",
 "status": "ACTIVE",
 "error": "ACCESS_DENIED",
 "createdAt": "2022-07-14T01:31:43.104000+00:00",
 "resourceType": "AWS::KMS::Key",
 "findingType": "ExternalAccess",
 "resourceOwnerAccount": "941407043048",
 "analyzedAt": "2025-03-19T06:51:46.090000+00:00",
 "id": "7ef6f04a-9d2c-4038-9cc0-2a5f00a4d8f8",
 "updatedAt": "2022-07-14T01:31:43.104000+00:00"
}
```


Résultats d'erreur des accès internes

Les analyseurs d'accès internes, qui identifient les accès au sein de votre compte ou de votre organisation, peuvent générer quatre types de résultats d'erreur :

- PRINCIPAL_LIMIT_EXCEEDED : générée lorsque plus de 3 000 principaux ont accès à une ressource critique. Cette erreur vous aide à identifier les ressources dont l'accès est trop large et qu'il peut être nécessaire de restreindre.

  Si vous apportez des modifications à la ressource ou aux principaux dans votre environnement qui font passer le nombre de principaux en dessous de la limite, l'analyseur générera des résultats normaux lors de la prochaine analyse, et le résultat d'erreur sera marqué comme résolu.


```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "PRINCIPAL_LIMIT_EXCEEDED",
```

```
 "resource": "arn:aws:s3:::critical-data",
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
 "findingType": "InternalAccess",
 "findingDetails": [
  {
   "internalAccessDetails": {}
  }
 ]
}
```

- Erreurs au niveau des ressources (INTERNAL_ERROR ou ACCESS_DENIED) : tout comme les erreurs d'accès externes, elles indiquent que l'analyseur n'a pas pu analyser une ressource spécifique en raison de problèmes internes ou d'autorisations. Lorsqu'une erreur au niveau d'une ressource se produit, l'analyseur génère un seul résultat d'erreur pour la ressource au lieu des résultats normaux.

```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "INTERNAL_ERROR", // can be INTERNAL_ERROR or ACCESS_DENIED
 "resource": "arn:aws:s3:::critical-data",
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
 "findingType": "InternalAccess",
 "findingDetails": [
  {
   "internalAccessDetails": {}
  }
 ]
}
```

- Erreurs au niveau du principal (INTERNAL_ERROR ou ACCESS_DENIED) : indique que l'analyseur n'a pas pu analyser l'accès d'un principal spécifique à une ressource spécifique. Contrairement aux erreurs au niveau des ressources, une ressource peut présenter à la fois des résultats normaux pour certains principaux et des résultats d'erreur pour d'autres principaux.

```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "INTERNAL_ERROR", // can be INTERNAL_ERROR or ACCESS_DENIED
 "resource": "arn:aws:s3:::critical-data",
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
 "findingType": "InternalAccess",
 "findingDetails": [
  {
   "internalAccessDetails": {
    "principal": {
     "AWS": "arn:aws:iam::111122223333:role/MyRole_1"
    },
    "principalOwnerAccount": "111122223333",
    "principalType": "IAM_ROLE",
    "accessType": "INTRA_ACCOUNT"
   }
  }
 ]
}
```

- PRINCIPAL_ERRORS_LIMIT_EXCEDED : générée lorsqu'un trop grand nombre d'erreurs de niveau principal sont détectées pour une seule ressource. Il s'agit d'une erreur au niveau des ressources qui peut apparaître parallèlement aux résultats normaux pour la même ressource.

```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "PRINCIPAL_ERRORS_LIMIT_EXCEEDED",
 "resource": "arn:aws:s3:::critical-data",
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
 "findingType": "InternalAccess",
 "resourceControlPolicyRestriction": "NOT_APPLICABLE",
 "serviceControlPolicyRestriction": "NOT_APPLICABLE",
```

```
 "findingDetails": [
  {
   "internalAccessDetails": {}
  }
 ]
}
```

Résolution des résultats d’erreur

Si vous résolvez le problème qui a empêché l’analyseur d’accès IAM d’analyser la ressource, le résultat d’erreur est complètement supprimé au lieu d’être transformé en résultat résolu.

Pour résoudre les résultats d’erreur, envisagez les approches suivantes en fonction du type d’erreur :

- Pour les erreurs ACCESS_DENIED, vérifiez que le rôle lié au service d’analyse d’accès IAM dispose des autorisations nécessaires pour accéder à la ressource.
- Pour les erreurs PRINCIPAL_LIMIT_EXCEEDED, vérifiez les stratégies d’accès à la ressource et envisagez de restreindre l’accès à un nombre plus restreint de principaux.
- Pour les résultats d'INTERNAL_ERROR, vous devrez peut-être attendre un cycle d'analyse suivant ou contacter le AWS support si le problème persiste.
- Pour PRINCIPAL_ERRORS_LIMIT_EXCEEDED, examinez et simplifiez éventuellement les modèles d’accès pour la ressource concernée.

Après avoir apporté les modifications nécessaires pour résoudre les problèmes sous-jacents, l’analyseur d’accès IAM tentera d’analyser à nouveau les ressources lors de son prochain cycle d’analyse.

Types de ressources prises en charge par l’analyseur d’accès IAM pour l’accès externe et interne

Pour les analyseurs d'accès externes et internes, IAM Access Analyzer analyse les politiques basées sur les ressources qui sont appliquées aux AWS ressources de la région dans laquelle vous avez activé IAM Access Analyzer. Seules les politiques basées sur les ressources sont analysées. Pour plus d’informations sur la manière dont l’analyseur d’accès IAM génère des résultats pour chaque type de ressource, consultez les informations relatives aux types de ressources.

> **Note**
>
> Les types de ressources pris en charge répertoriés concernent les analyseurs d'accès externes et internes. Les analyseurs d'accès internes ne prennent pas en charge tous les types de ressources pris en charge par les analyseurs d'accès externes. Les analyseurs d'accès non utilisés ne prennent en charge que les utilisateurs et les rôles IAM. Pour de plus amples informations, veuillez consulter Fonctionnement des résultats de l'analyseur d'accès IAM.

Types de ressources pris en charge pour l'accès externe

- Compartiments Amazon Simple Storage Service
- Compartiments du répertoire Amazon Simple Storage Service
- Gestion des identités et des accès AWS rôles
- AWS Key Management Service clés
- AWS Lambda fonctions et couches
- Files d'attente Amazon Simple Queue Service
- AWS Secrets Manager secrets
- Rubriques Amazon Simple Notification Service
- Instantanés volumes Amazon Elastic Block Store
- Instantanés de base de données service de base de données relationnelle Amazon
- Instantanés de cluster de base de données service base de données relationnelle Amazon
- Référentiels Amazon Elastic Container Registry
- Systèmes de fichiers Amazon Elastic File System
- Amazon DynamoDB Streams
- Tables Amazon DynamoDB

Types de ressources pris en charge pour l'accès interne

- Compartiments Amazon Simple Storage Service
- Compartiments du répertoire Amazon Simple Storage Service

- Instantanés de base de données service de base de données relationnelle Amazon
- Instantanés de cluster de base de données service base de données relationnelle Amazon
- Amazon DynamoDB Streams
- Tables Amazon DynamoDB

Compartiments Amazon Simple Storage Service

Lorsque l'analyseur d'accès IAM analyse les compartiments Amazon S3 pour les analyseurs d'accès externes, il génère un résultat lorsqu'une stratégie de compartiment Amazon S3, une liste de contrôle d'accès (ACL) ou un point d'accès, y compris un point d'accès multi-régions, appliqué à un compartiment accorde l'accès à une entité externe. Une entité externe est un principal ou une autre entité que vous pouvez utiliser pour créer un filtre qui ne se trouve pas dans votre zone d'approbation. Par exemple, si une politique de compartiment accorde l'accès à un autre compte ou autorise un accès public, IAM Access Analyzer génère un résultat. Toutefois, si vous activez le blocage de l'accès public sur votre compartiment, vous pouvez bloquer l'accès au niveau du compte ou du compartiment.

Pour les analyseurs d'accès internes, l'analyseur d'accès IAM génère un résultat lorsqu'un principal (utilisateur ou rôle) au sein de votre organisation ou de votre compte a accès à un compartiment Amazon S3 spécifié.

> **Note**
>
> IAM Access Analyzer n'analyse pas la politique de point d'accès associée aux points d'accès intercompte, car le point d'accès et sa politique ne font pas partie du compte de l'analyseur. IAM Access Analyzer génère un résultat public lorsqu'un compartiment délègue l'accès à un point d'accès intercompte et que l'option Bloquer l'accès public n'est pas activée sur le compartiment ou le compte. Lorsque vous activez Bloquer l'accès public, le résultat public est résolu et IAM Access Analyzer génère un résultat intercompte pour le point d'accès intercompte.

Les paramètres Amazon S3 de blocage de l'accès public remplacent les politiques de compartiment qui sont appliquées au compartiment. Les paramètres remplacent également les politiques de point d'accès appliquées aux points d'accès du compartiment. Chaque fois qu'une politique change, IAM Access Analyzer analyse les paramètres de blocage de l'accès public au niveau du compartiment. Cependant, il évalue les paramètres de blocage d'accès public au niveau du compte seulement une

fois toutes les 6 heures. Cela signifie que l’IAM Access Analyzer peut ne pas générer ou résoudre un résultat pour un accès public à un compartiment pendant une période pouvant aller jusqu'à 6 heures. Par exemple, si vous disposez d'une politique de compartiment qui autorise l'accès public, IAM Access Analyzer génère un résultat pour cet accès. Si vous activez le blocage de l'accès public afin de bloquer tout accès public au compartiment au niveau du compte, IAM Access Analyzer ne résout pas le résultat pour la politique de compartiment pendant une période pouvant aller jusqu'à 6 heures, même si tout l'accès public au compartiment est bloqué. La résolution des résultats publics concernant les points d'accès intercompte peut également prendre jusqu'à 6 heures une fois que vous avez activé l'option Bloquer l'accès public au niveau du compte. Les modifications apportées à une politique de contrôle des ressources (RCP) sans modification de politique de compartiment n’entraînent pas une nouvelle analyse du compartiment mentionné dans le résultat. L’analyseur d’accès IAM analyse la politique nouvelle ou mise à jour au cours de l’analyse périodique suivante, qui a lieu dans un délai maximal de 24 heures.

Pour un point d'accès multi-région, IAM Access Analyzer utilise une politique établie dans le but de générer des résultats. IAM Access Analyzer évalue les modifications apportées aux points d'accès multi-région toutes les 6 heures. Cela signifie que l’IAM Access Analyzer ne génère pas ou ne résout pas un résultat pendant au moins 6 heures, même si vous créez ou supprimez un point d'accès multi-région, ou si vous mettez à jour la politique pour ce point.

Compartiments du répertoire Amazon Simple Storage Service

Les compartiments de répertoires Amazon S3 organisent les données de manière hiérarchique dans des répertoires, contrairement à la structure de stockage horizontale des compartiments à usage général, ce qui est recommandé pour les charges de travail ou les applications où les performances sont essentielles. Pour les analyseurs d’accès externes, l’analyseur d’accès IAM analyse la stratégie de compartiments de répertoires, y compris les instructions de condition dans une politique, qui permettent à une entité externe d’accéder à un compartiment de répertoires.

Pour les analyseurs d’accès internes, l’analyseur d’accès IAM génère un résultat lorsqu’un principal (utilisateur ou rôle) au sein de votre organisation ou de votre compte a accès à un compartiment de répertoire Amazon S3 spécifié.

Les compartiments de répertoires Amazon S3 prennent également en charge les points d’accès, qui appliquent des autorisations et des contrôles réseau distincts pour toutes les demandes adressées au compartiment de répertoires via le point d’accès. Chaque point d’accès peut avoir une politique de point d’accès personnalisée qui fonctionne conjointement avec la stratégie de compartiment associée au compartiment du répertoire sous-jacent. Grâce aux points d’accès pour les compartiments de

répertoire, vous pouvez restreindre l'accès à des préfixes spécifiques, à des actions API ou à un cloud privé virtuel (VPC).

> **Note**
>
> IAM Access Analyzer n'analyse pas la politique de point d'accès associée aux points d'accès intercompte, car le point d'accès et sa politique ne font pas partie du compte de l'analyseur. IAM Access Analyzer génère un résultat public lorsqu'un compartiment délègue l'accès à un point d'accès intercompte et que l'option Bloquer l'accès public n'est pas activée sur le compartiment ou le compte. Lorsque vous activez Bloquer l'accès public, le résultat public est résolu et IAM Access Analyzer génère un résultat intercompte pour le point d'accès intercompte.

Pour plus d'informations sur les compartiments de répertoire Amazon S3, consultez Utilisation des compartiments de répertoire dans le guide de l'utilisateur Amazon Simple Storage Service.

Gestion des identités et des accès AWS rôles

Pour les rôles IAM, IAM Access Analyzer analyse les politiques de confiance. Dans une politique d'approbation de rôle, vous définissez les principaux auxquels vous faites confiance pour endosser le rôle. Une politique d'approbation de rôle est une politique basée sur les ressources requise qui est attachée à un rôle dans IAM. IAM Access Analyzer génère des résultats pour les rôles dans la zone de confiance à laquelle peut accéder une entité externe située en dehors de votre zone de confiance.

> **Note**
>
> Un rôle IAM est une ressource globale. Si une politique d'approbation de rôle accorde l'accès à une entité externe, IAM Access Analyzer génère un résultat dans chaque région activée.

AWS Key Management Service clés

En AWS KMS keys effet, IAM Access Analyzer analyse les politiques clés et les autorisations appliquées à une clé. IAM Access Analyzer génère un résultat si une politique de clé ou d' un octroi autorise l' accès d'une entité externe à la clé. Par exemple, si vous utilisez la clé de CallerAccount condition kms : dans une déclaration de politique pour autoriser l'accès à tous les utilisateurs d'un AWS compte spécifique, et que vous spécifiez un compte autre que le compte courant (la zone de

confiance de l'analyseur actuel), IAM Access Analyzer génère un résultat. Pour en savoir plus sur les clés de AWS KMS condition dans les déclarations de politique IAM, consultez la section Clés de AWS KMS condition.

Lorsque l'IAM Access Analyzer analyse une clé KMS, il lit les métadonnées de clé, telles que la politique de clé et la liste des autorisations. Si la politique de clé n'autorise pas le rôle IAM Access Analyzer à lire les métadonnées de clé, un résultat d'erreur de type Accès refusé est généré. Par exemple, si l'instruction de politique suivante est la seule politique appliquée à une clé, un résultat d'erreur de type Accès refusé est généré dans IAM Access Analyzer.

```
{
    "Sid": "Allow access for Key Administrators",
    "Effect": "Allow",
    "Principal": {
       "AWS": "arn:aws:iam::111122223333:role/Admin"
    },
    "Action": "kms:*",
    "Resource": "*"
}
```

Comme cette instruction autorise uniquement le rôle nommé Admin du AWS compte 111122223333 à accéder à la clé, un message d'erreur d'accès refusé est généré car IAM Access Analyzer n'est pas en mesure d'analyser complètement la clé. Un résultat d'erreur s'affiche sous la forme d'un texte en rouge dans le tableau Findings (Résultats). Le résultat ressemble à ce qui suit :

```
{
    "error": "ACCESS_DENIED",
    "id": "12345678-1234-abcd-dcba-111122223333",
    "analyzedAt": "2019-09-16T14:24:33.352Z",
    "resource": "arn:aws:kms:us-
west-2:1234567890:key/1a2b3c4d-5e6f-7a8b-9c0d-1a2b3c4d5e6f7g8a",
    "resourceType": "AWS::KMS::Key",
    "status": "ACTIVE",
    "updatedAt": "2019-09-16T14:24:33.352Z"
}
```

Lorsque vous créez une clé KMS, les autorisations accordées pour accéder à la clé dépendent de la façon dont vous créez celle-ci. Si vous recevez un résultat d'erreur de type Accès refusé pour une ressource de clé, appliquez l'instruction de politique suivante à la ressource de clé pour accorder à IAM Access Analyzer l'autorisation d'accéder à la clé.

```
{
    "Sid": "Allow IAM Access Analyzer access to key metadata",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/aws-service-role/access-
analyzer.amazonaws.com/AWSServiceRoleForAccessAnalyzer"
        },
    "Action": [
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms:List*"
    ],
    "Resource": "*"
},
```

Après la réception d'un résultat d'erreur de type Accès refusé pour une ressource de clé KMS, puis la résolution du résultat par la mise à jour de la politique de clé, le résultat est mis à jour et prend le statut Resolved (Résolu). S'il existe des instructions de politique ou des autorisations de clé qui accordent l'autorisation sur la clé à une entité externe, vous pouvez voir des résultats supplémentaires pour la ressource clé.

AWS Lambda fonctions et couches

Pour les AWS Lambda fonctions, IAM Access Analyzer analyse les politiques, y compris les déclarations de condition figurant dans une politique, qui accordent l'accès à la fonction à une entité externe. Avec Lambda, vous pouvez associer des politiques uniques basées sur les ressources aux fonctions, aux versions, aux alias et aux couches. L'analyseur d'accès IAM signale les accès externes en fonction des politiques basées sur les ressources associées aux fonctions et aux couches. L'analyseur d'accès IAM ne signale pas les accès externes sur la base de politiques basées sur les ressources associées aux alias et aux versions spécifiques invoquées à l'aide d'un ARN qualifié.

Pour plus d'informations, consultez les sections Utilisation de politiques basées sur les ressources pour Lambda et Utilisation de versions dans le Guide du développeur. AWS Lambda

Files d'attente Amazon Simple Queue Service

Pour les files d'attente Amazon SQS, IAM Access Analyzer analyse les politiques, notamment les instructions de condition dans une politique, permettant à une entité externe l'accès à une file d'attente.

AWS Secrets Manager secrets

Pour détecter AWS Secrets Manager les secrets, IAM Access Analyzer analyse les politiques, y compris les déclarations de condition figurant dans une stratégie, qui permettent à une entité externe d'accéder à un secret.

Rubriques Amazon Simple Notification Service

IAM Access Analyzer analyse les politiques basées sur les ressources associées aux rubriques Amazon SNS, y compris les conditions énoncées dans les politiques qui autorisent l'accès externe à une rubrique. Vous pouvez autoriser les comptes externes à effectuer des actions Amazon SNS, telles que l'abonnement à des sujets et la publication de sujets, par le biais d'une politique basée sur les ressources. Une rubrique Amazon SNS est accessible de l'extérieur si les responsables d'un compte situé en dehors de votre zone de confiance peuvent effectuer des opérations sur le sujet. Lorsque vous choisissez `Everyone` dans votre politique lors de la création d'un sujet Amazon SNS, vous le rendez accessible au public. `AddPermission` est une autre façon d'ajouter une politique basée sur les ressources à une rubrique Amazon SNS qui autorise l'accès externe.

Instantanés volumes Amazon Elastic Block Store

Les instantanés volume d'Amazon Elastic Block Store n’ont pas de politique basée sur les ressources. Un instantané est partagé via les autorisations de partage Amazon EBS. Pour les instantanés volume Amazon EBS, IAM Access Analyzer analyse les listes de contrôle d'accès qui permettent à une entité externe d'accéder à un instantané. Un instantané d'un volume Amazon EBS peut être partagé avec des comptes externes lorsqu'il est crypté. Un instantané de volume non chiffré peut être partagé avec des comptes externes et accorder un accès public. Les paramètres de partage se trouvent dans l'attribut `CreateVolumePermissions` de l'instantané. Lorsque les clients prévisualisent l'accès externe à un instantané Amazon EBS, ils peuvent spécifier la clé de chiffrement pour indiquer que l'instantané est chiffré, de la même manière qu'IAM Access Analyzer Preview gère les secrets de Secrets Manager.

Instantanés de base de données service de base de données relationnelle Amazon

Les Instantanés de base de données Amazon RDS ne disposent pas de politiques basées sur les ressources. Un instantané de base de données est partagé via les autorisations de base de données Amazon RDS, et seuls les instantanés de base de données manuels peuvent être partagés. Pour les analyseurs d’accès externes, l’analyseur d’accès IAM analyse les listes de contrôle d’accès qui permettent à une entité externe d’accéder à un instantané de base de données Amazon RDS.

Les instantanés de base de données non cryptés peuvent être publics. Les instantanés de base de données chiffrés ne peuvent pas être partagés publiquement, mais ils peuvent être partagés avec jusqu'à 20 autres comptes. Pour de plus amples informations, veuillez consulter Création d'un instantané de base de données. IAM Access Analyzer considère la possibilité d'exporter un instantané manuel de base de données (par exemple, vers un compartiment Amazon S3) comme un accès sécurisé.

Pour les analyseurs d'accès internes, l'analyseur d'accès IAM génère un résultat lorsqu'un principal (utilisateur ou rôle) au sein de votre organisation ou de votre compte a accès à un instantané de base de données Amazon RDS spécifié.

> **Note**
>
> IAM Access Analyzer n'identifie pas les accès publics ou l'accès-intercompte configurés directement dans la base de données elle-même. IAM Access Analyzer identifie uniquement les résultats relatifs à l'accès public ou l'accès-intercompte configuré sur l'instantané de base de données Amazon RDS.

Instantanés de cluster de base de données service base de données relationnelle Amazon

Les instantanés de cluster de base de données Amazon RDS ne disposent pas de politiques basées sur les ressources. Un instantané est partagé via les autorisations du cluster de base de données Amazon RDS. Pour les analyseurs d'accès externes, l'analyseur d'accès IAM analyse les listes de contrôle d'accès qui permettent à une entité externe d'accéder à un instantané du cluster de la base de données Amazon RDS. Les instantanés de cluster non chiffrés peuvent être publics. Les instantanés de cluster chiffrés ne peuvent pas être partagés publiquement. Les instantanés de cluster chiffrés et non chiffrés et peuvent être partagés avec 20 autres comptes maximum. Pour plus d'informations, consultez Creating a DB Cluster Snapshot (Créer un instantané de cluster de base de données). IAM Access Analyzer considère la possibilité d'exporter un instantané de cluster de bases de données (par exemple, vers un compartiment Amazon S3) comme un accès sécurisé.

Pour les analyseurs d'accès internes, l'analyseur d'accès IAM génère un résultat lorsqu'un principal (utilisateur ou rôle) au sein de votre organisation ou de votre compte a accès à un instantané du cluster de la base de données Amazon RDS spécifié.

> **Note**
>
> Les résultats d'IAM Access Analyzer n'incluent pas la surveillance d'une partie des clusters et clones de bases de données Amazon RDS utilisés par un autre Compte AWS utilisateur ou une organisation. AWS Resource Access Manager IAM Access Analyzer identifie uniquement les résultats relatifs à l'accès public ou accès-intercompte configuré sur l'instantané du cluster de base de données Amazon RDS.

Référentiels Amazon Elastic Container Registry

Pour les référentiels Amazon ECR, IAM Access Analyzer analyse les politiques basées sur les ressources, y compris les instructions de condition dans une politique, qui permettent à une entité externe d'accéder à un référentiel (comme les autres types de ressources tels que les rubriques Amazon SNS et les systèmes de fichiers d’EFS Amazon SNS). Pour les référentiels Amazon ECR, un principal doit être autorisé à `ecr:GetAuthorizationToken` via une politique basée sur l’identité pour être considéré comme disponible en externe.

Systèmes de fichiers Amazon Elastic File System

Pour les systèmes de fichiers Amazon EFS, IAM Access Analyzer analyse les politiques, notamment les instructions de condition dans une politique, permettant à une entité externe l’accès à un système de fichiers. Un système de fichiers Amazon EFS est accessible de l'extérieur si les responsables d'un compte situé en dehors de votre zone de confiance peuvent effectuer des opérations sur ce système de fichiers. L'accès est défini par une politique de système de fichiers qui utilise IAM et par la manière dont le système de fichiers est monté. Par exemple, le montage de votre système de fichiers Amazon EFS sur un autre compte est considéré comme accessible de l'extérieur, sauf si ce compte appartient à votre organisation et que vous l'ayez définie comme votre zone de confiance. Si vous montez le système de fichiers à partir d'un cloud privé virtuel avec un sous-réseau public, le système de fichiers est accessible de l'extérieur. Lorsque vous utilisez Amazon EFS avec AWS Transfer Family, les demandes d'accès au système de fichiers reçues d'un serveur Transfer Family appartenant à un compte différent de celui du système de fichiers sont bloquées si le système de fichiers autorise l'accès public.

Amazon DynamoDB Streams

Pour les analyseurs d’accès externes, l’analyseur d’accès IAM génère un résultat si une politique DynamoDB autorise au moins une action intercompte permettant à une entité externe d’accéder

à un flux DynamoDB. Pour plus d'informations sur les actions intercomptes prises en charge pour DynamoDB, consultez Actions IAM supportées par des politiques basées sur les ressources dans le guide du développeur Amazon DynamoDB.

Pour les analyseurs d'accès internes, l'analyseur d'accès IAM génère un résultat lorsqu'un principal (utilisateur ou rôle) au sein de votre organisation ou de votre compte a accès à un flux DynamoDB spécifié.

Tables Amazon DynamoDB

Pour les analyseurs d'accès externes, l'analyseur d'accès IAM génère un résultat pour une table DynamoDB si une politique DynamoDB autorise au moins une action intercompte permettant à une entité externe d'accéder à une table ou à un index DynamoDB. Pour plus d'informations sur les actions intercomptes prises en charge pour DynamoDB, consultez Actions IAM supportées par des politiques basées sur les ressources dans le guide du développeur Amazon DynamoDB.

Pour les analyseurs d'accès internes, l'analyseur d'accès IAM génère un résultat lorsqu'un principal (utilisateur ou rôle) au sein de votre organisation ou de votre compte a accès à une table DynamoDB spécifiée.

Administrateur délégué pour IAM Access Analyzer

Si vous configurez AWS Identity and Access Management Access Analyzer dans votre compte AWS Organizations de gestion, vous pouvez ajouter un compte de membre de l'organisation en tant qu'administrateur délégué chargé de gérer IAM Access Analyzer pour votre organisation. L'administrateur délégué dispose des autorisations pour créer et gérer des analyseurs au sein de l'organisation. Seul le compte de gestion peut ajouter un administrateur délégué.

L'administrateur délégué pour l'analyseur d'accès IAM est un compte membre de l'organisation qui dispose d'autorisations pour créer et gérer des analyseurs qui analysent l'accès au sein de l'organisation. Seul le compte de gestion peut ajouter, supprimer ou modifier un administrateur délégué.

Si vous ajoutez un administrateur délégué, vous pouvez ultérieurement passer à un compte différent pour l'administrateur délégué. Dans ce cas, le compte d'administrateur délégué précédent perd l'autorisation sur tous les analyseurs qui ont été créés à l'aide de ce compte pour analyser les accès à travers l'organisation. Ces analyseurs passent à l'état désactivé et ne génèrent plus de résultats et ne mettent pas à jour les résultats existants. Les résultats existants pour ces analyseurs ne sont plus accessibles. Vous pouvez y accéder à nouveau ultérieurement en configurant le compte en

tant qu'administrateur délégué. Si vous savez que vous n'utiliserez pas le même compte qu'un administrateur délégué, vous pouvez envisager de supprimer les analyseurs avant de changer d'administrateur délégué. Cela supprime tous les résultats générés. Lorsque le nouvel administrateur délégué crée de nouveaux analyseurs, de nouvelles instances des mêmes résultats sont générées. Vous ne perdez pas les résultats. Ceux-ci sont simplement générés pour le nouvel analyseur dans un compte différent. Vous pouvez également continuer à accéder aux résultats de l'organisation à l'aide du compte de gestion de l'organisation, qui dispose également des autorisations d'administrateur. Le nouvel administrateur délégué doit créer de nouveaux analyseurs pour que l'IAM Access Analyzer commence à surveiller les ressources de votre organisation.

Si l'administrateur délégué quitte l' AWS organisation, les privilèges d'administration délégués sont supprimés du compte. Tous les analyseurs du compte dont l'organisation est la zone de confiance passent à l'état désactivé. Les résultats existants pour ces analyseurs ne sont plus accessibles.

La première fois que vous configurez des analyseurs dans le compte de gestion, vous pouvez choisir l'option Ajouter un administrateur délégué sur la page Paramètres de l'analyseurdans la console de l'analyseur d'accès IAM.

> **Note**
>
> L'analyseur d'accès IAM facture les analyseurs d'accès non utilisés en fonction du nombre de rôles et d'utilisateurs IAM analysés par analyseur et par mois. Si vous créez un analyseur d'accès non utilisé dans le compte de gestion et le compte d'administrateur délégué, les deux analyseurs d'accès non utilisés vous seront facturés. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

Après avoir modifié l'administrateur délégué, le nouvel administrateur doit créer des analyseurs pour commencer à surveiller l'accès aux ressources de votre organisation.

Ajout d'un administrateur délégué pour l'analyseur d'accès IAM

Si vous configurez AWS Identity and Access Management Access Analyzer dans votre compte AWS Organizations de gestion, vous pouvez ajouter un compte membre de l'organisation en tant qu'administrateur délégué chargé de gérer IAM Access Analyzer pour votre organisation. L'administrateur délégué dispose des autorisations pour créer et gérer des analyseurs au sein de l'organisation. Seul le compte de gestion peut ajouter un administrateur délégué.

Pour ajouter un administrateur délégué à l'aide de la console

1. Connectez-vous à la AWS console à l'aide du compte de gestion de votre organisation.
2. Ouvrez la console IAM à l’adresse https://console.aws.amazon.com/iam/.
3. Sous Accéder à l’analyseur, sélectionnez Paramètres de l’analyseur.
4. Choisissez Add delegated administrator (Ajouter un administrateur délégué).
5. Dans le champ Administrateur délégué, saisissez le numéro de Compte AWS d’un compte membre de l’organisation à désigner comme administrateur délégué.

   Le compte doit être membre de votre organisation.
6. Sélectionnez Enregistrer les modifications.

Pour ajouter un administrateur délégué à l'aide du AWS CLI ou du AWS SDKs

Lorsque vous créez un analyseur pour analyser l'accès au sein de l'organisation dans un compte d'administrateur délégué à l'aide de la AWS CLI, de l' AWS API (à l'aide de AWS SDKs) ou CloudFormation, vous devez l'utiliser AWS Organizations APIs pour activer l'accès au service pour IAM Access Analyzer et enregistrer le compte du membre en tant qu'administrateur délégué.

1. Activez un accès aux services fiable pour IAM Access Analyzer dans. AWS Organizations Consultez la section Comment activer ou désactiver l'accès sécurisé dans le guide de AWS Organizations l'utilisateur.
2. Enregistrez un compte membre valide de votre AWS organisation en tant qu'administrateur délégué à l'aide de l'opération AWS Organizations `RegisterDelegatedAdministrator`API ou de la `register-delegated-administrator` AWS CLI commande.

Règles d'archivage

Les règles d'archivage archivent automatiquement les nouvelles découvertes qui répondent aux critères que vous définissez lors de la création de la règle. Vous pouvez également appliquer des règles d'archivage rétroactivement pour archiver les résultats existants qui répondent aux critères des règles d'archivage. Par exemple, vous pouvez créer une règle d’archivage pour archiver automatiquement tous les résultats pour un compartiment Amazon S3 spécifique auquel vous accordez régulièrement l’accès. Si vous accordez à un principal spécifique l'accès à plusieurs ressources, vous pouvez créer une règle qui archive automatiquement tout nouveau résultat généré

pour l'accès accordé à ce principal. Cela vous permet de vous concentrer uniquement sur les résultats actifs susceptibles d'indiquer un risque de sécurité.

Lorsque vous créez une règle d'archivage, seules les nouvelles découvertes correspondant aux critères de la règle sont automatiquement archivées. Les résultats existants ne sont pas automatiquement archivés. Lorsque vous créez une règle, vous pouvez inclure jusqu'à 20 valeurs par critère dans la règle. Pour obtenir la liste des clés de filtre que vous pouvez utiliser pour créer ou mettre à jour une règle d'archivage, veuillez consulter Clés de filtre de l'IAM Access Analyzer.

> **Note**
>
> Lorsque vous créez ou modifiez une règle d'archivage, l'IAM Access Analyzer ne valide pas les valeurs que vous incluez dans le filtre de la règle. Par exemple, si vous ajoutez une règle pour correspondre à un Compte AWS, l'analyseur d'accès IAM accepte toute valeur dans le champ, même s'il ne s'agit pas d'un numéro de compte AWS valide.

Pour créer une règle d'archivage

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Accéder à l'analyseur, puis sélectionnez Paramètres de l'analyseur.
3. Dans la section Analyseurs, choisissez l'analyseur pour lequel vous souhaitez créer une règle d'archivage.
4. Dans l'onglet Règles d'archivage, choisissez Créer une règle d'archivage.
5. Entrez un nom pour la règle si vous souhaitez modifier le nom par défaut.
6. Dans la section Rule (Règle) sous Criteria (Critères), sélectionnez une propriété à faire correspondre pour la règle.
7. Choisissez une condition pour la valeur de la propriété, telle que Contient, Est ou N'est pas égal à.

   Les opérateurs disponibles dépendent de la propriété que vous sélectionnez.
8. Vous pouvez également ajouter des valeurs supplémentaires pour la propriété ou ajouter des critères supplémentaires pour la règle. En ce qui concerne les résultats des accès externes, pour vous assurer que votre règle n'archive pas de nouveaux résultats en accès public, vous pouvez également inclure le critère Accès public et lui attribuer la valeur False.

Pour ajouter une valeur supplémentaire à un critère, sélectionnez Add another value (Ajouter une autre valeur). Pour ajouter un autre critère à la règle, choisissez Ajouter un critère.

9. Lorsque vous avez terminé d'ajouter des critères et des valeurs, sélectionnez Create rule (Créer une règle) pour appliquer la règle uniquement aux nouveaux résultats. Choisissez Create and archive active findings (Créer et archiver les résultats actifs) pour archiver les résultats nouveaux et existants en fonction des critères de la règle. Dans la section Results (Résultats), vous pouvez consulter la liste des résultats actifs auxquels la règle d'archivage s'applique.

Par exemple, pour créer une règle pour les résultats des accès externes qui archive automatiquement les résultats pour les compartiments Amazon S3 : sélectionnez Type de ressource, puis Est pour la condition. Choisissez ensuite le compartiment S3 dans la liste Valeur.

Pour créer une règle pour les résultats des accès non utilisés qui archive automatiquement tout résultat pour un compte particulier, choisissez Compte propriétaire de la ressource, puis Égal à pour la condition. Entrez l' Compte AWS ID dans la zone de texte Valeur.

Continuez à définir des critères pour personnaliser la règle en fonction de votre environnement, puis sélectionnez Créer une règle.

Si vous créez une nouvelle règle et ajoutez plusieurs critères, vous pouvez supprimer un seul critère de la règle en choisissant Supprimer ce critère. Vous pouvez supprimer une valeur ajoutée pour un critère en choisissant Remove value (Supprimer la valeur).

Pour modifier une règle d'archivage

1. Choisissez le nom de la règle à modifier dans la colonne Nom.

   Vous ne pouvez modifier qu'une seule règle d'archivage à la fois.
2. Ajoutez les nouveaux critères ou supprimez des critères et des valeurs existants pour chaque critère.
3. Choisissez Enregistrer les modifications pour appliquer la règle uniquement aux nouveaux résultats. Choisissez Enregistrer et archiver les résultats actifs pour archiver les résultats nouveaux et existants en fonction des critères de la règle.

Pour supprimer une règle d'archivage

1. Cochez la case correspondant aux règles que vous souhaitez supprimer.

2. Sélectionnez Delete (Supprimer).
3. Tapez **`delete`** dans la boîte de dialogue de confirmation Supprimer une règle d'archivage puis sélectionnez Supprimer.

Les règles sont supprimées uniquement de l'analyseur dans la région actuelle. Vous devez supprimer les règles d'archivage séparément pour chaque analyseur que vous avez créé dans d'autres régions.

Surveillance AWS Identity and Access Management Access Analyzer avec Amazon EventBridge

Utilisez les informations de cette rubrique pour savoir comment surveiller les résultats d'IAM Access Analyzer et accéder aux aperçus avec Amazon. EventBridge EventBridge est la nouvelle version d'Amazon CloudWatch Events.

Événements de résultats

IAM Access Analyzer envoie un événement EventBridge pour chaque résultat généré, pour une modification du statut d'un résultat existant et lorsqu'un résultat est supprimé. Pour recevoir les résultats et les notifications concernant les résultats, vous devez créer une règle d'événement dans Amazon EventBridge. Lorsque vous créez une règle d'événement, vous pouvez également spécifier une action cible à déclencher en fonction de la règle. Par exemple, vous pouvez créer une règle d'événement qui déclenche une rubrique Amazon SNS lorsqu'un événement pour un nouveau résultat est reçu en provenance de l’IAM Access Analyzer Les détails de la politique de contrôle des ressources (RCP) sont disponibles dans la section des détails des événements.

Événements de prévisualisation de l'accès

IAM Access Analyzer envoie un événement pour chaque aperçu de l'accès et EventBridge pour chaque modification de son statut. Cela inclut un événement lorsque la prévisualisation de l'accès est initialement créée (statut Création), lorsque la prévisualisation de l'accès est terminée (statut Terminé) ou lorsque la création de la prévisualisation de l'accès a échoué (statut Échec). Pour recevoir des notifications concernant les aperçus d'accès, vous devez créer une règle d'événement dans EventBridge. Lorsque vous créez une règle d'événement, vous pouvez spécifier une action cible à déclencher en fonction de la règle. Par exemple, vous pouvez créer une règle d'événement qui déclenche une rubrique Amazon SNS lorsqu'un événement pour une nouvelle prévisualisation est reçue en provenance de l’IAM Access Analyzer

Fréquence de notification d'événement

IAM Access Analyzer envoie des événements pour les nouvelles découvertes et les nouvelles découvertes avec des mises à jour de statut EventBridge dans l'heure qui suit la survenue de l'événement dans votre compte. IAM Access Analyzer envoie également des événements EventBridge lorsqu'un résultat résolu est supprimé en raison de l'expiration de la période de rétention. Pour les résultats supprimés parce que l'analyseur qui les a générés est supprimé, l'événement est envoyé EventBridge environ 24 heures après la suppression de l'analyseur. Lorsqu'un résultat est supprimé, le statut du résultat n'est pas modifié. En revanche, l'attribut `isDeleted` est défini sur `true`. IAM Access Analyzer envoie également des événements pour les aperçus d'accès nouvellement créés et les modifications du statut des aperçus d'accès à. EventBridge

Exemples d’événements liés aux résultats des accès externes

Voici un exemple d'événement de recherche d'accès externe IAM Access Analyzer envoyé à. EventBridge La `id` liste indique l'identifiant de l'événement dans EventBridge. Pour en savoir plus, consultez la section Événements et modèles d'événements dans EventBridge.

Dans l'objet `detail`, les valeurs des attributs `accountId` et `region` se réfèrent au compte et à la région indiqués dans le résultat. L'attribut `isDeleted` indique si l'événement provient du résultat en cours de suppression. L'attribut `id` représente l'ID de résultat. Le tableau `resources` est un singleton avec l'ARN de l'analyseur qui a généré le résultat.

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "action": [
            "s3:GetObject"
        ],
        "analyzedAt": "2019-11-21T01:22:22Z",
        "condition": {},
        "createdAt": "2019-11-20T04:58:50Z",
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "isPublic": false,
        "principal": {
            "AWS": "999988887777"
        },
        "region": "us-west-2",
```

```
        "resource": "arn:aws:s3:::amzn-s3-demo-bucket",
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
    "version": "0"
}
```

IAM Access Analyzer envoie également des événements EventBridge pour détecter les erreurs. Un résultat d'erreur est un résultat généré lorsque l'IAM Access Analyzer ne peut pas analyser la ressource. Les événements pour les résultats d'erreur incluent un attribut `error` comme illustré dans l'exemple suivant.

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "analyzedAt": "2019-11-21T01:22:22Z",
        "createdAt": "2019-11-20T04:58:50Z",
        "error": "ACCESS_DENIED",
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "region": "us-west-2",
        "resource": "arn:aws:s3:::amzn-s3-demo-bucket",
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
```

```
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
    "version": "0"
}
```

Exemples d'événements liés aux résultats des accès internes

Voici un exemple d'événement de recherche d'accès interne d'IAM Access Analyzer envoyé à. EventBridge La `id` liste indique l'identifiant de l'événement dans EventBridge. Pour en savoir plus, consultez la section Événements et modèles d'événements dans EventBridge.

Dans l'objet `detail`, les valeurs des attributs `accountId` et `principalOwnerAccount` se réfèrent au compte du principal indiqué dans le résultat. L'attribut `isDeleted` indique si l'événement provient du résultat en cours de suppression. L'attribut `id` représente l'ID de résultat. Le `resource` est l'ARN de l'analyseur qui a généré le résultat.

```
{
    "version": "0",
    "id": "b45c3678-c278-b593-6121-c155259ce1b5",
    "detail-type": "Internal Access Finding",
    "source": "aws.access-analyzer",
    "account": "111122223333",
    "time": "2025-04-08T19:42:49Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:access-analyzer:us-east-1:111122223333:analyzer/testAnalyzer"
    ],
    "detail": {
        "accessType": "INTRA_ACCOUNT",
        "action": [
            "s3:GetObject"
        ],
        "analyzedAt": "2025-04-08T03:18:43.509465073Z",
        "condition": {},
        "createdAt": "2025-04-07T21:33:49.914099224Z",
        "id": "11111111-2222-4444-aaaa-333333333333",
        "isDeleted": false,
        "findingType": "InternalAccess",
        "principal": {
            "AWS": "arn:aws:iam::111122223333:role/MyRole_6"
        },
```

```
        "principalOwnerAccount": "111122223333",
        "principalType": "IAM_ROLE",
        "resource": "arn:aws:s3:::critical-data",
        "resourceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "accountId": "111122223333",
        "resourceType": "AWS::S3::Bucket",
        "serviceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "status": "ACTIVE",
        "updatedAt": "2025-04-08T03:22:12.654688231Z",
        "version": "1.0"
    }
}
```

IAM Access Analyzer envoie également des événements EventBridge pour détecter les erreurs. Un résultat d'erreur est un résultat généré lorsque l'IAM Access Analyzer ne peut pas analyser la ressource. Les événements pour les résultats d'erreur incluent un attribut `error` comme illustré dans l'exemple suivant.

```
{
    "version": "0",
    "id": "5a94b99b-e87d-a6a7-58c7-f47871532860",
    "detail-type": "Internal Access Finding",
    "source": "aws.access-analyzer-test",
    "account": "444455556666",
    "time": "2025-05-07T11:57:54Z",
    "region": "us-west-2",
    "resources": ["arn:aws:access-analyzer-beta:us-west-2:444455556666:analyzer/
example-analyzer"],
    "detail": {
        "analyzedAt": "2025-03-24T19:58:52.512329448Z",
        "createdAt": "2025-03-22T03:30:46.920200692Z",
        "id": "ef573afd-12a5-4095-87a6-bf2f25109895",
        "isDeleted": false,
        "findingType": "InternalAccess",
        "resource": "arn:aws:s3:::test-entity-88",
        "accountId": "111122223333",
        "resourceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "resourceType": "AWS::S3::Bucket",
        "serviceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "error": "ACCESS_DENIED", // can be INTERNAL_ERROR and ACCESS_DENIED
        "status": "ACTIVE",
        "updatedAt": "2025-03-24T20:09:39.176075014Z",
        "version": "1.0"
```

```
    }
}
```

Exemples d'événements liés aux résultats des accès non utilisés

Voici un exemple d'événement de recherche d'accès non utilisé d'IAM Access Analyzer envoyé à. EventBridge La `id` liste indique l'identifiant de l'événement dans EventBridge. Pour en savoir plus, consultez la section Événements et modèles d'événements dans EventBridge.

Dans l'objet `detail`, les valeurs des attributs `accountId` et `region` se réfèrent au compte et à la région indiqués dans le résultat. L'attribut `isDeleted` indique si l'événement provient du résultat en cours de suppression. L'attribut `id` représente l'ID de résultat.

```
{
    "version": "0",
    "id": "dc7ce3ee-114b-3243-e249-7f10f9054b21",
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
    "account": "123456789012",
    "time": "2023-09-29T17:31:40Z",
    "region": "us-west-2",
    "resources": [
       "arn:aws:access-analyzer:us-west-2:123456789012:analyzer/
integTestLongLivingAnalyzer-DO-NOT-DELETE"
       ],
    "detail": {
        "findingId": "b8ae0460-5d29-4922-b92a-ba956c986277",
        "resource": "arn:aws:iam::111122223333:role/FindingIntegTestFakeRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "111122223333",
        "createdAt": "2023-09-29T17:29:18.758Z",
        "updatedAt": "2023-09-29T17:29:18.758Z",
        "analyzedAt": "2023-09-29T17:29:18.758Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "62160bda-8e94-46d6-ac97-9670930d8ffb",
        "isDeleted": false,
        "findingType": "UnusedPermission",
        "numberOfUnusedServices": 0,
        "numberOfUnusedActions": 1
        }
    }
```

IAM Access Analyzer envoie également des événements EventBridge pour détecter les erreurs. Un résultat d'erreur est un résultat généré lorsque l'IAM Access Analyzer ne peut pas analyser la ressource. Les événements pour les résultats d'erreur incluent un attribut `error` comme illustré dans l'exemple suivant.

```
{
    "version": "0",
    "id": "c2e7aa1a-4df7-7652-f33e-64113b8997d4",
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
    "account": "111122223333",
    "time": "2023-10-31T20:26:12Z",
    "region": "us-west-2",
    "resources": [
      "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/ba811f91-
de99-41a4-97c0-7481898b53f2"
      ],
    "detail": {
        "findingId": "b01a34f2-e118-46c9-aef8-0d8526b495c7",
        "resource": "arn:aws:iam::123456789012:role/TestRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "444455556666",
        "createdAt": "2023-10-31T20:26:08.647Z",
        "updatedAt": "2023-10-31T20:26:09.245Z",
        "analyzedAt": "2023-10-31T20:26:08.525Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "7c7a72a2-7963-4c59-ac71-f0be597010f7",
        "isDeleted": false,
        "findingType": "UnusedIAMRole",
        "error": "INTERNAL_ERROR"
        }
  }
```

Exemple d'événement de prévisualisation de l'accès

L'exemple suivant montre les données du premier événement envoyé EventBridge lorsque vous créez un aperçu des accès. Le tableau `resources` est un singleton avec l'ARN de l'analyseur auquel la prévisualisation de l'accès est associée. Dans l'objet `detail`, l'`id` fait référence à l'ID de prévisualisation de l'accès et `configuredResources` fait référence à la ressource pour laquelle la prévisualisation de l'accès a été créée. Le `status` est `Creating` et fait référence au statut de la

prévisualisation de l'accès. Le `previousStatus` n'est pas spécifié car la prévisualisation de l'accès vient juste d'être créée.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.00Z",
        "region": "us-west-2",
        "status": "CREATING",
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "aaaabbbb-2222-3333-4444-555566667777",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

L'exemple suivant montre les données d'un événement envoyé à EventBridge pour un aperçu de l'accès avec un changement de statut de `Creating` à`Completed`. Dans l'objet détail, l'ID `id` fait référence à l'ID de prévisualisation de l'accès. Le `status` et le `previousStatus` font référence au statut de la prévisualisation de l'accès, où le statut précédent était `Creating` et où le statut actuel est `Completed`.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.000Z",
        "previousStatus": "CREATING",
        "region": "us-west-2",
```

```
        "status": "COMPLETED",
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "11112222-3333-4444-5555-666677778888",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

L'exemple suivant montre les données d'un événement envoyé à EventBridge pour un aperçu de l'accès avec un changement de statut de `Creating` à`Failed`. Dans l'objet `detail`, l'`id` fait référence à l'ID de prévisualisation de l'accès. Le `status` et le `previousStatus` font référence au statut de la prévisualisation de l'accès, où le statut précédent était `Creating` et où le statut actuel est `Failed`. Le champ `statusReason` fournit le code raison indiquant que la prévisualisation de l'accès a échoué en raison d'une configuration de ressource non valide.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.00Z",
        "previousStatus": "CREATING",
        "region": "us-west-2",
        "status": "FAILED",
        "statusReason": {
            "code": "INVALID_CONFIGURATION"
        },
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "99998888-7777-6666-5555-444433332222",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
```

```
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

Création d'une règle d'événement à l'aide de la console

La procédure suivante explique comment créer une règle d'événement à l'aide de la console.

1. Ouvrez la EventBridge console Amazon à l'adresse https://console.aws.amazon.com/events/.
2. À l'aide des valeurs suivantes, créez une EventBridge règle qui surveille la recherche d'événements ou l'accès aux événements de prévisualisation :
   - Pour Type de règle, choisissez Règle avec un modèle d'événement.
   - Pour Event source (Source de l'événement), choisissez Other (Autres).
   - Pour Event pattern (Modèle d'événement), choisissez Custom patterns (JSON editor) (Modèle personnalisé [éditeur JSON]), puis collez l'un des exemples de modèle d'événement suivant dans la zone de texte :
     - Pour créer une règle basée sur n'importe quel événement de l'analyseur d'accès IAM, utilisez l'exemple de modèle suivant :

       ```
       {
         "source": [
           "aws.access-analyzer"
         ]
       }
       ```

     - Pour créer une règle basée sur un événement de résultats des accès externes, internes ou non utilisés, utilisez l'exemple de modèle suivant :

       ```
       {
         "source": [
           "aws.access-analyzer"
         ],
         "detail-type": [
           "Access Analyzer Finding",
           "Internal Access Finding",
           "Unused Access Finding for IAM entities"
         ]
       ```

```
}
```

- Pour créer une règle basée sur un événement de résultats des accès externes, utilisez l'exemple de modèle suivant :

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Analyzer Finding"
  ]
}
```

- Pour créer une règle basée sur un événement de résultats des accès internes, utilisez l'exemple de modèle suivant :

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Internal Access Finding"
  ]
}
```

- Pour créer une règle basée sur un événement de résultats des accès non utilisés, utilisez l'exemple de modèle suivant :

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Unused Access Finding for IAM entities"
  ]
}
```

- Pour créer une règle basée sur un événement de prévisualisation de l'accès, utilisez l'exemple de modèle suivant :

```
{
```

```
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Preview State Change"
  ]
}
```

- Pour les types de cible, choisissez un AWS service, et pour Sélectionner une cible, choisissez une cible telle qu'un sujet ou AWS Lambda une fonction Amazon SNS. La cible est déclenchée lorsqu'un événement correspond au modèle d'événement défini dans la règle est reçu.

Pour en savoir plus sur la création de règles, consultez la section Création de EventBridge règles Amazon qui réagissent aux événements dans le guide de EventBridge l'utilisateur Amazon.

Création d'une règle d'événement à l'aide de la CLI

1. Utilisez ce qui suit pour créer une règle pour Amazon EventBridge à l'aide du AWS CLI. Remplacez le nom de la règle *TestRule* par le nom de votre règle.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-analyzer\"]}"
```

2. Vous pouvez personnaliser la règle pour déclencher des actions cibles uniquement pour un sous-ensemble de résultats générés, tels que les résultats avec des attributs spécifiques. L'exemple suivant montre comment créer une règle qui déclenche une action cible uniquement pour les résultats ayant un statut actif.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-analyzer\"],\"detail-type\":[\"Access Analyzer Finding\"],\"detail\":{\"status\":[\"ACTIVE\"]}}"
```

L'exemple suivant montre comment créer une règle qui déclenche une action cible uniquement pour les prévisualisations d'accès ayant un statut qui passe de `Creating` à `Completed`.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"],\"detail-type\":[\"Access Preview State Change\"],\"detail\":{\"status
\":[\"COMPLETED\"]}}"
```

3. Pour définir une fonction Lambda en tant que cible pour la règle que vous avez créée, utilisez l'exemple de commande suivant. Remplacez la région et le nom de la fonction dans l'ARN en fonction de votre environnement.

```
aws events put-targets --rule TestRule --targets Id=1,Arn=arn:aws:lambda:us-
east-1:111122223333:function:MyFunction
```

4. Ajoutez les autorisations requises pour appeler la cible de la règle. L'exemple suivant montre comment accorder des autorisations à une fonction Lambda, en suivant les exemples précédents.

```
aws lambda add-permission --function-name MyFunction --statement-id 1 --action
 'lambda:InvokeFunction' --principal events.amazonaws.com
```

Intégrez IAM Access Analyzer à AWS Security Hub CSPM

AWS Security Hub CSPM fournit une vue complète de votre état de sécurité dans AWS. Il vous aide à évaluer votre environnement par rapport aux normes de sécurité du secteur et aux pratiques exemplaires. Security Hub CSPM collecte des données de sécurité provenant de l'ensemble Comptes AWS des services et des produits partenaires tiers pris en charge. Il analyse ensuite vos tendances en matière de sécurité et identifie les problèmes de sécurité les plus prioritaires.

Lorsque vous intégrez IAM Access Analyzer à Security Hub CSPM, vous pouvez envoyer les résultats d'IAM Access Analyzer au Security Hub CSPM. Security Hub CSPM peut ensuite inclure ces résultats dans son analyse de votre posture de sécurité globale.

Table des matières

- Comment IAM Access Analyzer envoie ses résultats à Security Hub (CSPM)
  - Types de résultats envoyés par l'analyseur d'accès IAM
  - Latence pour l'envoi des résultats
  - Réessayer lorsque Security Hub CSPM n'est pas disponible
  - Mise à jour des résultats existants dans Security Hub CSPM

- Affichage des résultats de l'analyseur d'accès IAM dans Security Hub CSPM
  - Interprétation de la recherche de noms par IAM Access Analyzer dans Security Hub CSPM
- Résultats typiques de l’analyseur d’accès IAM
- Activation et configuration de l'intégration
- Comment arrêter l'envoi des résultats

Comment IAM Access Analyzer envoie ses résultats à Security Hub (CSPM)

Dans Security Hub CSPM, les problèmes de sécurité sont suivis en tant que findings (résultats). Certains résultats proviennent de problèmes détectés par d'autres partenaires Services AWS ou par des partenaires tiers. Security Hub CSPM dispose également d'un ensemble de règles qu'il utilise pour détecter les problèmes de sécurité et générer des résultats.

Security Hub CSPM fournit des outils permettant de gérer les résultats provenant de toutes ces sources. Vous pouvez afficher et filtrer les listes de résultats et afficher les informations relatives à chaque résultat. Pour de plus amples informations, consultez la section Viewing findings (Affichage des résultats) dans le Guide de l'utilisateur AWS Security Hub . Vous pouvez également suivre le statut des analyses dans les résultats. Pour de plus amples informations, veuillez consulter Prendre des mesure en fonction des résultats dans le Guide de l'utilisateur AWS Security Hub .

Tous les résultats du Security Hub CSPM utilisent un format JSON standard appelé AWS Security Finding Format (ASFF). Le format ASFF comprend des informations sur la source du problème, les ressources affectées et le statut actuel du résultat. Pour de plus amples informations, veuillez consulter AWS Security Finding Format (ASFF) dans le Guide de l'utilisateur AWS Security Hub .

AWS Identity and Access Management Access Analyzer est l'un des outils Services AWS qui envoie les résultats au Security Hub CSPM. Pour les accès non utilisés, l’analyseur d’accès IAM détecte les accès non utilisés accordés aux utilisateurs ou aux rôles IAM et génère un résultat pour chacun d’entre eux. IAM Access Analyzer envoie ensuite ces résultats à Security Hub CSPM.

Pour l’accès externe, l’analyseur d’accès IAM détecte les instructions de politique qui autorisent l’accès public ou l’accès intercompte aux principaux externes sur les ressources prises en charge dans votre organisation ou votre compte.??? IAM Access Analyzer génère une recherche d'accès public et l'envoie à Security Hub CSPM. Pour l'accès entre comptes, IAM Access Analyzer envoie un résultat unique pour un principal externe à la fois à Security Hub CSPM. S'il existe plusieurs résultats entre comptes dans IAM Access Analyzer, vous devez résoudre le résultat du Security Hub CSPM pour le principal externe unique avant qu'IAM Access Analyzer ne fournisse le résultat croisé suivant. Pour une liste complète des principaux externes disposant d’un accès intercompte en dehors de la

zone confiance de l'analyseur, vous devez consulter les résultats dans l'analyseur d'accès IAM. Les détails de la politique de contrôle des ressources (RCP) sont disponibles dans la section des détails de la ressource.

Types de résultats envoyés par l'analyseur d'accès IAM

IAM Access Analyzer envoie les résultats à Security Hub CSPM en utilisant le format ASFF (AWS Security Finding Format). Dans le format ASFF, le champ `Types` fournit le type de résultat. Les résultats de l'analyseur d'accès IAM peuvent avoir les valeurs suivantes pour `Types`.

- Résultats relatifs à l'accès externe — Effects/Data Exposition/accès externe accordé
- Résultats relatifs à l'accès externe — Vérifications du logiciel et de la configuration/Meilleur accès en AWS matière de sécurité accordé Practices/External
- Résultats d'accès non utilisés : vérifications du logiciel et de la configuration/meilleure autorisation AWS de sécurité Practices/Unused
- Résultats d'accès non utilisés : vérifications du logiciel et de la configuration/meilleur Practices/Unused rôle IAM en AWS matière de sécurité
- Résultats d'accès non utilisés — Vérifications du logiciel et de la AWS configuration/sécurité Meilleur mot de passe utilisateur Practices/Unused IAM
- Résultats d'accès non utilisés — Vérifications du logiciel et de la AWS configuration/sécurité Meilleure clé d'accès Practices/Unused utilisateur IAM

Latence pour l'envoi des résultats

Lorsque IAM Access Analyzer crée une nouvelle découverte, elle est généralement envoyée au Security Hub CSPM dans les 30 minutes. Toutefois, dans de rares cas, l'analyseur d'accès IAM peut ne pas être informé d'un changement de politique. Par exemple :

- Les modifications des paramètres de blocage de l'accès public Amazon S3 au niveau du compte peuvent prendre jusqu'à 12 heures pour être répercutées dans l'analyseur d'accès IAM.
- Les modifications apportées à une politique de contrôle des ressources (RCP) sans modification de politique basée sur une ressource n'entraînent pas une nouvelle analyse de la ressource mentionnée dans le résultat. L'analyseur d'accès IAM analyse la politique nouvelle ou mise à jour au cours de l'analyse périodique suivante, qui a lieu dans un délai maximal de 24 heures.
- En cas de problème de livraison lié à la livraison du AWS CloudTrail journal, un changement de politique risque de ne pas déclencher une nouvelle analyse de la ressource signalée dans le résultat.

En pareil cas, l’analyseur d’accès IAM analyse la politique nouvelle ou mise à jour au cours de l’analyse périodique suivante.

Réessayer lorsque Security Hub CSPM n'est pas disponible

Si Security Hub CSPM n'est pas disponible, IAM Access Analyzer essaie à nouveau d'envoyer les résultats régulièrement.

Mise à jour des résultats existants dans Security Hub CSPM

Après avoir envoyé un résultat à Security Hub CSPM, IAM Access Analyzer continue d'envoyer des mises à jour pour refléter toute observation supplémentaire concernant l'activité de recherche à Security Hub CSPM. Ces mises à jour sont mentionnées dans le même résultat.

Pour les résultats des accès externes, l’analyseur d’accès IAM les regroupe par ressource. Dans Security Hub CSPM, la recherche d'une ressource reste active si au moins l'une des conclusions relatives à cette ressource est active dans IAM Access Analyzer. Si tous les résultats d'IAM Access Analyzer relatifs à une ressource sont archivés ou résolus, les résultats du Security Hub CSPM sont également archivés. Le résultat du Security Hub CSPM est mis à jour lorsque la politique d'accès change entre un accès public et un accès entre comptes. Cette mise à jour peut inclure des modifications du type, du titre, de la description et de la gravité du résultat.

Pour les résultats d’accès non utilisés, l’analyseur d’accès IAM ne les regroupe pas par ressource. En revanche, si un résultat d'accès non utilisé est résolu dans IAM Access Analyzer, le résultat correspondant du Security Hub CSPM est également résolu. Le résultat CSPM du Security Hub est mis à jour lorsque vous mettez à jour l'utilisateur ou le rôle IAM qui a généré le résultat d'accès non utilisé.

Affichage des résultats de l'analyseur d'accès IAM dans Security Hub CSPM

Pour consulter les résultats de votre analyse d'accès IAM dans Security Hub CSPM, choisissez Voir les résultats dans la section AWS: Analyseur d'accès IAM de la page de résumé. Vous pouvez également choisir Findings (Résultats) dans le panneau de navigation. Vous pouvez ensuite filtrer les résultats pour n'afficher que AWS Identity and Access Management Access Analyzer les résultats en choisissant le champ Nom du produit : avec une valeur de**`IAM Access Analyzer`**.

Interprétation de la recherche de noms par IAM Access Analyzer dans Security Hub CSPM

AWS Identity and Access Management Access Analyzer envoie les résultats à Security Hub CSPM en utilisant le format ASFF (AWS Security Finding Format). Dans ASFF, le champ Types fournit le type de recherche. Les types ASFF utilisent un schéma de dénomination différent de AWS Identity

and Access Management Access Analyzer. Le tableau suivant contient des informations détaillées sur tous les types ASFF associés aux AWS Identity and Access Management Access Analyzer résultats tels qu'ils apparaissent dans Security Hub CSPM.

Type de résultat ASFF	Security Hub CSPM trouve un titre	Description
Effects/Data Exposure/ ExternalAccès accordé	<ARN de la ressource >permet l'accès public	Une politique basée sur les ressources attachée à la ressource permet à touts les principaux externes d'accéder à la ressource.
Vérifications du logiciel et de la configuration/Meilleur accès en AWS matière de sécurité accordé Practices/External	<ARN de la ressource >permet l'accès entre comptes	Une politique basée sur les ressources attachée à la ressource fournit un accès entre comptes aux principaux externes en dehors de la zone confiance de l'analyseur .
Vérifications du logiciel et de la configuration/Bonnes pratiques de AWS sécurité/ Autorisations non utilisées	<resource ARN> contient des autorisations non utilisées	Un utilisateur ou un rôle contient des autorisations de service et d’action non utilisés.
Contrôles du logiciel et de la configuration/meilleur Practices/Unused rôle IAM en AWS matière de sécurité	<resource ARN> contient un rôle IAM non utilisé	Un utilisateur ou un rôle contient un rôle IAM non utilisé.
Vérifications du logiciel et de la AWS configuration/sécurité Meilleur mot de passe utilisate ur Practices/Unused IAM	<resource ARN> contient un mot de passe utilisateur IAM non utilisé	Un utilisateur ou un rôle contient un mot de passe utilisateur IAM non utilisé.
Contrôles du logiciel et de la AWS configuration/sécurité	<resource ARN> contient la clé d’accès utilisateur IAM non utilisée	Un utilisateur ou un rôle contient une clé d’accès utilisateur IAM non utilisée.

Type de résultat ASFF	Security Hub CSPM trouve un titre	Description
Meilleure clé d'accès Practices /Unused utilisateur IAM		

Résultats typiques de l'analyseur d'accès IAM

IAM Access Analyzer envoie ses résultats au Security Hub CSPM en utilisant le format ASFF (AWS Security Finding Format).

Voici un exemple de résultat typique de l'analyseur d'accès IAM pour les résultats d'accès externes.


```
{
    "SchemaVersion": "2018-10-08",
    "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/my-analyzer/
arn:aws:s3:::amzn-s3-demo-bucket",
    "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
    "GeneratorId": "aws/access-analyzer",
    "AwsAccountId": "111122223333",
    "Types": ["Software and Configuration Checks/AWS Security Best Practices/External
 Access Granted"],
    "CreatedAt": "2020-11-10T16:17:47Z",
    "UpdatedAt": "2020-11-10T16:43:49Z",
    "Severity": {
        "Product": 1,
        "Label": "LOW",
        "Normalized": 1
    },
    "Title": "AwsS3Bucket/arn:aws:s3:::amzn-s3-demo-bucket/ allows cross-account
 access",
    "Description": "AWS::S3::Bucket/arn:aws:s3:::amzn-s3-demo-bucket/ allows cross-
account access from AWS 444455556666",
    "Remediation": {
        "Recommendation": {"Text": "If the access isn't intended, it indicates a
 potential security risk. Use the console for the resource to modify or remove the
 policy that grants the unintended access. You can use the Rescan button on the Finding
 details page in the Access Analyzer console to confirm whether the change removed the
 access. If the access is removed, the status changes to Resolved."}
    },
    "SourceUrl": "https://console.aws.amazon.com/access-analyzer/home?region=us-
west-2#/findings/details/dad90d5d-63b4-6575-b0fa-ef9c556ge798",
```

```
    "Resources": [
        {
            "Type": "AwsS3Bucket",
            "Id": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Details": {
                "Other": {
                    "External Principal Type": "AWS",
                    "Condition": "none",
                    "Action Granted": "s3:GetObject,s3:GetObjectVersion",
                    "External Principal": "444455556666"
                }
            }
        }
    ],
    "WorkflowState": "NEW",
    "Workflow": {"Status": "NEW"},
    "RecordState": "ACTIVE"
}
```

Voici un exemple de résultat typique de l'analyseur d'accès IAM pour les résultats des accès non utilisés.

```
{
    "Findings": [
    {
      "SchemaVersion": "2018-10-08",
      "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-
DO-NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
      "ProductName": "IAM Access Analyzer",
      "CompanyName": "AWS",
      "Region": "us-west-2",
      "GeneratorId": "aws/access-analyzer",
      "AwsAccountId": "111122223333",
      "Types": [
        "Software and Configuration Checks/AWS Security Best Practices/Unused
 Permission"
      ],
      "CreatedAt": "2023-09-18T16:29:09.657Z",
      "UpdatedAt": "2023-09-21T20:39:16.651Z",
      "Severity": {
        "Product": 1,
        "Label": "LOW",
```

```
        "Normalized": 1
      },
      "Title": "AwsIamRole/arn:aws:iam::111122223333:role/IsengardRole-DO-NOT-DELETE/
 contains unused permissions",
      "Description": "AWS::IAM::Role/arn:aws:iam::111122223333:role/IsengardRole-DO-
NOT-DELETE/ contains unused service and action-level permissions",
      "Remediation": {
        "Recommendation": {
          "Text":"If the unused permissions aren't required, delete the permissions to
 refine access to your account. Use the IAM console to modify or remove the policy that
 grants the unused permissions. If all the unused permissions are removed, the status
 of the finding changes to Resolved."
        }
      },
      "SourceUrl": "https://us-west-2.console.aws.amazon.com/access-analyzer/
home?region=us-west-2#/unused-access-findings?resource=arn%3Aaws%3Aiam%3A
%3A903798373645%3Arole%2FTestRole",
      "ProductFields": {
      "numberOfUnusedActions": "256",
      "numberOfUnusedServices": "15",
      "resourceOwnerAccount": "111122223333",
      "findingId": "DEMO24d8d-0d3f-4d3d-99f4-299fc8a62ee7",
      "findingType": "UnusedPermission",
      "aws/securityhub/FindingId": "arn:aws:securityhub:us-west-2::product/aws/access-
analyzer/arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-DO-
NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "aws/securityhub/ProductName": "AM Access Analyzer",
      "aws/securityhub/CompanyName": "AWS"
    },
    "Resources": [
    {
      "Type": "AwsIamRole",
      "Id": "arn:aws:iam::111122223333:role/TestRole"
    }
  ],
  "WorkflowState": "NEW",
  "Workflow": {
    "Status": "NEW"
    },
  "RecordState": "ARCHIVED",
  "FindingProviderFields": {
    "Severity": {
      "Label": "LOW"
    },
```

```
    "Types": [
    "Software and Configuration Checks/AWS Security Best Practices/Unused Permission"
    ]
  }
  }
]
}
```

Activation et configuration de l'intégration

Pour utiliser l'intégration avec Security Hub CSPM, vous devez activer Security Hub CSPM. Pour plus d'informations sur l'activation de Security Hub CSPM, consultez la section Configuration de Security Hub dans le guide de l'AWS Security Hub utilisateur.

Lorsque vous activez à la fois IAM Access Analyzer et Security Hub CSPM, l'intégration est automatiquement activée. IAM Access Analyzer commence immédiatement à envoyer ses résultats à Security Hub CSPM.

Comment arrêter l'envoi des résultats

Pour arrêter l'envoi des résultats à Security Hub CSPM, vous pouvez utiliser la console Security Hub CSPM ou l'API.

Veuillez consulter Désactivation et activation du flux de résultats à partir d'une intégration (console) ou Désactivation du flux de résultats d'une intégration (API Security Hub, AWS CLI) dans le Guide de l'utilisateur AWS Security Hub .

Journalisation des appels d'API IAM Access Analyzer avec AWS CloudTrail

IAM Access Analyzer est intégré à AWS CloudTrail un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans IAM Access Analyzer. CloudTrail capture tous les appels d'API pour IAM Access Analyzer sous forme d'événements. Les appels enregistrés contiennent les appels de la console IAM Access Analyzer ainsi que les appels de code aux opérations d'API IAM Access Analyzer.

Si vous créez un suivi, vous pouvez activer la diffusion continue d' CloudTrail événements vers un compartiment Amazon S3, y compris des événements pour IAM Access Analyzer. Si vous ne configurez pas de suivi, vous pouvez toujours consulter les événements les plus récents dans la CloudTrail console dans Historique des événements.

À l'aide des informations collectées par CloudTrail, vous pouvez déterminer la demande envoyée à IAM Access Analyzer, l'adresse IP à partir de laquelle la demande a été faite, l'auteur de la demande, la date à laquelle elle a été faite et des informations supplémentaires.

Pour en savoir plus CloudTrail, consultez le guide de AWS CloudTrail l'utilisateur.

Informations sur l'analyseur d'accès IAM dans CloudTrail

CloudTrail est activé sur votre AWS compte lorsque vous le créez. Lorsqu'une activité se produit dans IAM Access Analyzer, cette activité est enregistrée dans un CloudTrail événement avec d'autres événements de AWS service dans l'historique des événements. Vous pouvez consulter, rechercher et télécharger les événements récents dans votre AWS compte. Pour plus d'informations, consultez la section Affichage des événements avec l'historique des CloudTrail événements.

Pour un enregistrement continu des événements de votre AWS compte, y compris des événements pour IAM Access Analyzer, créez un suivi. Un suivi permet CloudTrail de fournir des fichiers journaux à un compartiment Amazon S3. Par défaut, lorsque vous créez un parcours dans la console, celui-ci s'applique à toutes les AWS régions. Le journal enregistre les événements de toutes les régions de la AWS partition et transmet les fichiers journaux au compartiment Amazon S3 que vous spécifiez. En outre, vous pouvez configurer d'autres AWS services pour analyser plus en détail les données d'événements collectées dans les CloudTrail journaux et agir en conséquence. Pour plus d'informations, consultez les ressources suivantes :

- Vue d'ensemble de la création d'un journal d'activité
- CloudTrail Services et intégrations pris en charge
- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers CloudTrail journaux de plusieurs régions et réception de fichiers CloudTrail journaux de plusieurs comptes

Toutes les actions d'IAM Access Analyzer sont enregistrées CloudTrail et documentées dans la référence d'API d'IAM Access Analyzer. Par exemple, les appels au `CreateAnalyzer` `CreateArchiveRule` et les `ListFindings` actions génèrent des entrées dans les fichiers CloudTrail journaux.

Chaque événement ou entrée de journal contient des informations sur la personne ayant initié la demande. Les informations relatives à l'identité permettent de déterminer les éléments suivants :

- Si la demande a été faite avec les informations d'identification de l'utilisateur root ou Gestion des identités et des accès AWS (IAM).
- Si la demande a été effectuée avec les informations d'identification de sécurité temporaires d'un rôle ou d'un utilisateur fédéré.
- Si la demande a été faite par un autre AWS service.

Pour plus d'informations, consultez la section Élément userIdentity CloudTrail .

Présentation des entrées des fichiers journaux IAM Access Analyzer

Un suivi est une configuration qui permet de transmettre des événements sous forme de fichiers journaux à un compartiment Amazon S3 que vous spécifiez. CloudTrail les fichiers journaux contiennent une ou plusieurs entrées de journal. Un événement représente une demande unique provenant de n'importe quelle source et inclut des informations sur l'action demandée, la date et l'heure de l'action, les paramètres de la demande, etc. CloudTrail les fichiers journaux ne constituent pas une trace ordonnée des appels d'API publics, ils n'apparaissent donc pas dans un ordre spécifique.

L'exemple suivant montre une entrée de CloudTrail journal qui illustre l'`CreateAnalyzer`opération effectuée par une session assumée nommée `Alice-tempcreds` le « 14 juin 2021 ». La séance de rôle a été générée par le rôle nommé `admin-tempcreds`.

```
{
  "eventVersion": "1.05",
  "userIdentity":   {
    "type": "AssumedRole",
    "principalId": "AROAIBKEVSQ6C2EXAMPLE:Alice-tempcreds",
    "arn": "arn:aws:sts::111122223333:assumed-role/admin-tempcreds/Alice-tempcreds",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "true",
        "creationDate": "2021-06-14T22:54:20Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:iam::111122223333:role/admin-tempcreds",
        "accountId": "111122223333",
```

```
        "userName": "admin-tempcreds"
      },
     "webIdFederationData": {},
    }
  },
  "eventTime": "2021-06-14T22:57:36Z",
  "eventSource": "access-analyzer.amazonaws.com",
  "eventName": "CreateAnalyzer",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "198.51.100.179",
  "userAgent": "aws-sdk-java/1.12.79 Linux/5.4.141-78.230 OpenJDK_64-
Bit_Server_VM/25.302-b08 java/1.8.0_302 vendor/Oracle_Corporation cfg/retry-mode/
standard",
  "requestParameters": {
    "analyzerName": "test",
    "type": "ACCOUNT",
    "clientToken": "11111111-abcd-2222-abcd-222222222222",
        "tags": {
            "tagkey1": "tagvalue1"
        }
  },
  "responseElements": {
    "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/test"
  },
  "requestID": "22222222-dcba-4444-dcba-333333333333",
  "eventID": "33333333-bcde-5555-bcde-444444444444",
  "readOnly": false,
  "eventType": "AwsApiCall",,
  "managementEvent": true,
  "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Clés de filtre de l'IAM Access Analyzer

Vous pouvez utiliser les clés de filtre ci-dessous pour définir une règle d'archivage (`CreateArchiveRule`), mettre à jour une règle d'archivage (`UpdateArchiveRule`), extraire une liste de résultats (`ListFindings` et `ListFindingsV2`) ou extraire une liste de résultats de prévisualisation d'accès pour une ressource (`ListAccessPreviewFindings`). Il n'y a aucune différence entre l'utilisation de l'API IAM et CloudFormation la configuration des règles d'archivage.

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
ressource	Ressourc	ARN identifiant de manière unique la ressource à laquelle le principal externe a accès. Pour en savoir plus, consultez les noms des ressources Amazon (ARNs).	String	Oui	Oui	Oui	Externe Internal (Interne) Non utilisé
resourceType `AWS::S3 Bucket | AWS::IAM :Role | AWS::SQS :Queue | AWS::Lambda::Function | AWS::Lambda::Layer`	Type de ressource	Type de ressource auquel le principal externe a accès. **Note** Les analyseurs d'accès internes ne prennent pas en charge tous les types de ressources pris en	String	Oui	Oui	Oui	Externe Internal (Interne) Non utilisé

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
`Version |AWS::KM :Key | AWS::Se etsMana r::Secr | AWS::EF :FileSy em | AWS::EC :Snapsh | AWS::EC :Reposi ry | AWS::RD :DBSnap ot | AWS::RD :DBClus rSnapsh | AWS::SN :Topic | AWS::S3 press::`		charge par les analyseurs d'accès externes. Les analyseurs d'accès non utilisés ne prennent en charge que les utilisateurs et les rôles IAM. Pour de plus amples informations, veuillez consulter Types de ressources prises en charge par l'analyseur d'accès					

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
`rectoryl cket | AWS::Dyı moDB::Tı le | AWS::Dyı moDB::S eam | AWS::IAI :User`		IAM pour l'accès externe et interne.					
resourceOwnerAccount	Compte du propriétaire de la ressource	L'identifiant de AWS compte à 12 chiffres propriétaire de la ressource. Pour en savoir plus, consultez Identifiants de compte AWS.	String	✓ Oui	✓ Oui	✓ Oui	Externe Internal (Interne) Non utilisé
isPublic	Accès public	Indique si le résultat signale une ressource dotée d'une politique autorisant l'accès public.	Booléen	✓ Oui	✓ Oui	✓ Oui	Externe

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
findingType `ExternalAccess` | `UnusedIAMRole` | `UnusedIAMUserAccessKey` | `UnusedIAMUserPassword` | `UnusedPermission` | `InternalAccess`	Types de résultats	Type du résultat . Pour les analyseurs d'accès externes, le type est `ExternalAccess` . Pour les analyseurs d'accès non utilisés, le type peut être `UnusedIAMRole` , `UnusedIAMUserAccessKey` , `UnusedIAMUserPassword` ou `UnusedPermission` . Pour les analyseurs d'accès internes, le type est `InternalAccess` .	String	✓ Oui	✓ Oui	✓ Oui	Externe Internal (Interne) Non utilisé

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
resourceControlPolicyRestriction `APPLIED | APPLICABLE | FAILED_TO_EVALUATE_RCP | NOT_APPLICABLE`	Restriction des politiques de contrôle des ressources (RCP)	Type de restriction appliqué par le propriétaire de la ressource avec une politique de contrôle des ressources (RCP) Organizations. Pour plus d'informations sur les valeurs de cette clé de filtre, consultez ExternalAccessDetailset InternalAccessDetailsdans le manuel de référence de l'API IAM Access Analyzer.	String	✓ Oui	✓ Oui	✓ Oui	Externe Internal (Interne)

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
serviceControlPolicyRestriction `APPLIED | APPLICABLE | FAILED_ _EVALUA _SCP | NOT_APPLICABLE`	Politique de contrôle des services (SCP)	Type de restriction appliqué par une politique de contrôle des services (SCP) Organizations. Pour plus d'informations sur les valeurs de cette clé de filtre, consultez le manuel InternalAccessDetailsde référence de l'API IAM Access Analyzer.	String	✓ Oui	✓ Oui	✓ Oui	Internal (Interne)
status `ACTIVE | ARCHIVED | RESOLVED`	Statut	Statut actuel du résultat.	String	✗ Non	✓ Oui	✓ Oui	Externe Internal (Interne) Non utilisé
error	Error (Erreur)	Indique l'erreur signalée pour le résultat.	String	✓ Oui	✓ Oui	✓ Oui	Externe Internal (Interne)

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
principal.AWS	AWS Compte	Le compte a accordé l'accès à la ressource dans le champ `Principal` du résultat. Entrez l'ID de AWS compte à 12 chiffres ou l'ARN de l' AWS utilisateur ou du rôle externe. Pour en savoir plus, consultez Identifiants de compte AWS.	String	✓ Oui	✓ Oui	✓ Oui	Externe
principal.Federated	Utilisateur fédéré	ARN de l'identité fédérée qui a accès à la ressource dans le résultat. Pour en savoir plus, veuillez consulter Fournisseurs d'identité et fédération.	String	✓ Oui	✓ Oui	✓ Oui	Externe

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
condition.aws : PrincipalArn	ARN principal	ARN du principal (utilisateur, rôle ou groupe IAM) indiquée comme condition d'accès aux ressources. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	String	✓ Oui	✓ Oui	✓ Oui	Externe
condition.aws : PrincipalOrgID	OrgID principal	Identifiant de l'organisation du principal indiqué comme condition d'accès aux ressources. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	String	✓ Oui	✓ Oui	✓ Oui	Externe

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
condition.aws : PrincipalOrgPaths	PrincipalOrgPaths	ID de l'organisation ou de l'unité organisationnelle (UO) indiqué comme condition d'accès aux ressources. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	String	✓ Oui	✓ Oui	✓ Oui	Externe
condition.aws : SourceIp	IP Source	Adresse IP qui autorise l'accès du principal à la ressource lors de l'utilisation de l'adresse IP spécifiée. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	Adresse IP	✓ Oui	✓ Oui	✓ Oui	Externe

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
condition.aws : SourceVp	Source VPC	ID de VPC qui autorise l'accès du principal à la ressource lors de l'utilisation du VPC spécifié. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	String	✓ Oui	✓ Oui	✓ Oui	Externe
condition.aws : UserId	ID de l'utilisateur	ID de l'utilisateur IAM à partir d'un compte externe indiqué comme condition d'accès à la ressource. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	String	✓ Oui	✓ Oui	✓ Oui	Externe

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
condition.aws : VpceAccount	Compte VPCE	ID du compte du point de terminaison d'un VPC qui permet au principal d'accéder à la ressource. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	String	✓ Oui	✓ Oui	✓ Oui	Externe Internal (Interne)
condition.aws : SourceVpcArn	Arne VPC source	L'ARN du VPC qui permet l'accès principal à la ressource lors de l'utilisation du VPC spécifié. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	ARN	✓ Oui	✓ Oui	✓ Oui	Externe

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
condition.aws : ID VpceOrg	OrgID VPCE	ID organisationnel du point de terminaison d'un VPC qui permet au principal d'accéder à la ressource. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	String	✓ Oui	✓ Oui	✓ Oui	Externe Internal (Interne)
condition.aws : VpceOrgPaths	VPCE OrgPaths	Unité organisationnelle (UO) du point de terminaison d'un VPC qui permet au principal d'accéder à la ressource. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	Chaîne (liste)	✓ Oui	✓ Oui	✓ Oui	Externe Internal (Interne)

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
condition.cognito-identity.amazonav.com:aud	Audience Cognito	ID de pool d'identités Amazon Cognito spécifié comme condition d'accès au rôle IAM dans le résultat. Pour en savoir plus, consultez les rubriques IAM et clés AWS STS contextuelles de condition.	String	✓ Oui	✓ Oui	✓ Oui	Externe

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
condition.graph.facebook.com:app_id	ID de l'application Facebook	ID d'application Facebook (ou ID de site) spécifié comme condition permettant d'autoriser l'accès de la fédération Login with Facebook au rôle IAM dans le résultat. Pour en savoir plus, consultez les rubriques IAM et clés AWS STS contextuelles de condition.	String	✓ Oui	✓ Oui	✓ Oui	Externe

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
condition.accounts.google.com:aud	Audience Google	ID d'application Google spécifié comme condition d'accès au rôle IAM. Pour en savoir plus, consultez les rubriques IAM et clés AWS STS contextuelles de condition.	String	✓ Oui	✓ Oui	✓ Oui	Externe

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
Condition. km : CallerAccount	ID de clé KMS	ID de AWS compte propriétaire de l'entité appelante (utilisateur IAM, rôle ou utilisateur root du compte) utilisée par les services qui appellent AWS KMS. Pour en savoir plus, consultez la section Clés de condition pour AWS Key Management Service.	String	✓ Oui	✓ Oui	✓ Oui	Externe

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
condition.www.amazon.com:app_id	ID de l'application Amazon	ID d'application Amazon (ou ID de site) spécifié comme condition permettant d'autoriser l'accès de la fédération Login with Amazon au rôle. Pour en savoir plus, veuillez consulter	String	✓ Oui	✓ Oui	✓ Oui	Externe
id	ID de résultat	ID du résultat.	String	✗ Non	✓ Oui	✓ Oui	Externe Internal (Interne) Non utilisé
changeType `CHANGED | NEW | UNCHANGE`		Donne un contexte sur la comparaison du résultat de la prévisualisation de l'accès à celui existant et identifié dans l'IAM Access Analyzer.	String	✗ Non	✗ Non	✓ Oui	Externe

Criterion	Console de gestion AWS champ	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès	Types d'analyseurs pris en charge
existingFindingId		L'ID existant du résultat dans l'IAM Access Analyzer, fourni uniquement pour les résultats trouvés dans la prévisualisation de l'accès.	String	⊗ Non	⊗ Non	✓ Oui	Externe
existingFindingStatus		Le statut existant du résultat, fourni uniquement pour les résultats existants dans la prévisualisation de l'accès.	String	⊗ Non	⊗ Non	✓ Oui	Externe

Utilisation de rôles liés à un service pour AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer utilise un rôle lié à un service IAM. Un rôle lié à un service est un type unique de rôle IAM lié directement à l'analyseur d'accès IAM. Les rôles liés aux services sont prédéfinis par IAM Access Analyzer et incluent toutes les autorisations requises par la fonctionnalité pour appeler d'autres AWS services en votre nom.

Un rôle lié à un service simplifie la configuration de l'analyseur d'accès IAM, car il n'est pas nécessaire d'ajouter manuellement les autorisations nécessaires. L'analyseur d'accès IAM définit

les autorisations des rôles liés à ses services et, sauf indication contraire, seul l'analyseur d'accès IAM peut assumer ces rôles. Les autorisations définies comprennent la politique d'approbation et la politique d'autorisation. De plus, cette politique d'autorisation ne peut pas être attachée à une autre entité IAM.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés à un service, veuillez consulter Services AWS qui fonctionnent avec IAM et recherchez les services où Oui figure dans la colonne Rôle lié à un service. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Autorisations de rôle liées à un service pour AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer utilise le rôle lié au service nommé AWSServiceRoleForAccessAnalyzer— Allow Access Analyzer pour analyser les métadonnées des ressources pour un accès externe et pour analyser l'activité afin d'identifier les accès non utilisés.

Le rôle AWSService RoleForAccessAnalyzer lié à un service fait confiance aux services suivants pour assumer le rôle :

- `access-analyzer.amazonaws.com`

La politique d'autorisations de rôle nommée `AccessAnalyzerServiceRolePolicy` autorise l'analyseur d'accès IAM à effectuer des actions sur des ressources spécifiques.

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour en savoir plus, consultez Service-Linked Role Permissions (autorisations du rôle lié à un service) dans le Guide de l'utilisateur IAM.

Création d'un rôle lié à un service pour l'analyseur d'accès IAM

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous activez Access Analyzer dans l'API Console de gestion AWS ou dans l' AWS API, IAM Access Analyzer crée le rôle lié au service pour vous. Le même rôle lié à un service est utilisé dans toutes les régions dans lesquelles vous activez IAM Access Analyzer. Les résultats des accès externes et non utilisés utilisent le même rôle lié au service.

> **Note**
>
> L'IAM Access Analyzer est régional. Vous devez activer l'IAM Access Analyzer dans chaque Région de manière indépendante.

Si vous supprimez ce rôle lié à un service, IAM Access Analyzer recrée le rôle lors de la prochaine création d'un analyseur.

Vous pouvez également utiliser la console IAM pour créer un rôle lié au service avec le cas d'utilisation Access Analyzer. Dans l'API AWS CLI ou dans l' AWS API, créez un rôle lié à un service avec le nom du `access-analyzer.amazonaws.com` service. Pour plus d'informations, consultez Création d'un rôle lié à un service dans le Guide de l'utilisateur IAM. Si vous supprimez ce rôle lié à un service, vous pouvez utiliser ce même processus pour créer le rôle à nouveau.

Modification d'un rôle lié à un service pour l'analyseur d'accès IAM

IAM Access Analyzer ne vous permet pas de modifier le rôle lié au AWSService RoleForAccessAnalyzer service. Après avoir créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour en savoir plus, consultez Modification d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Suppression d'un rôle lié à un service pour l'analyseur d'accès IAM

Si vous n'avez plus besoin d'utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, aucune entité inutilisée n'est surveillée ou gérée activement. Cependant, vous devez nettoyer les ressources de votre rôle lié à un service avant de pouvoir les supprimer manuellement.

Si IAM Access Analyzer est activé dans une ou plusieurs régions de votre organisation AWS Organizations, vous devez supprimer tous les analyseurs de toutes les régions de votre organisation avant de tenter de supprimer ce rôle.

> **Note**
>
> Si l'analyseur d'accès IAM utilise le rôle lorsque vous essayez de supprimer les ressources, alors la suppression peut échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour supprimer les ressources IAM Access Analyzer utilisées par le rôle AWSService RoleForAccessAnalyzer

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans la section Access reports (Rapports d'accès) sous Access Analyzer (Analyseur d'accès), sélectionnez Analyzers (Analyseurs).
3. Sélectionnez l'analyseur à partir duquel vous souhaitez supprimer les ressources de l'analyseur d'accès IAM associées au rôle lié au service.
4. Sélectionnez Delete (Supprimer).
5. Pour confirmer que vous souhaitez supprimer les analyseurs, entrez **`delete`**, puis sélectionnez Delete (Supprimer).

Pour supprimer manuellement le rôle lié au service à l'aide d'IAM

Utilisez la console IAM, le AWS CLI, ou l' AWS API pour supprimer le rôle lié au AWSService RoleForAccessAnalyzer service. Pour en savoir plus, consultez Suppression d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Régions prises en charge pour les rôles liés à un service de l'analyseur d'accès IAM

L'analyseur d'accès IAM prend en charge l'utilisation des rôles liés à un service dans toutes les régions où le service est disponible. Pour plus d'informations, consultez Régions et Points de terminaison AWS.

Prévisualiser l'accès

En plus de vous aider à identifier les ressources partagées avec une entité externe, l'analyseur d'accès AWS IAM vous permet de prévisualiser ses résultats avant de déployer des autorisations de ressources, de sorte à vérifier que vos modifications de politique n'octroient que l'accès public et intercompte prévu à votre ressource. Vous pouvez ainsi commencer avec l'accès externe prévu à vos ressources.

Vous pouvez prévisualiser et vérifier l'accès public et intercompte à vos compartiments Amazon S3 dans la console Amazon S3. Vous pouvez également utiliser les API IAM Access Analyzer pour prévisualiser l'accès public et inter-comptes à vos compartiments Amazon S3, à vos AWS KMS clés, à vos rôles IAM, à vos files d'attente Amazon SQS et à vos secrets Secrets Manager en proposant des autorisations pour votre ressource.

Rubriques

- Prévisualiser l'accès dans la console Amazon S3
- Prévisualisation de l'accès avec IAM Access Analyzer APIs

Prévisualiser l'accès dans la console Amazon S3

Une fois votre politique de compartiment réalisée dans la console Amazon S3, vous avez la possibilité de prévisualiser l'accès public et intercompte à votre compartiment Amazon S3. Vous pouvez vérifier que vos modifications de politique octroient uniquement l'accès externe prévu avant de choisir Save changes (Enregistrer les modifications). Cette étape facultative vous permet de prévisualiser les résultats AWS Identity and Access Management Access Analyzer pour votre compartiment. Vous pouvez vérifier si la modification de politique introduit de nouveaux résultats ou résout les résultats existants pour l'accès externe. Vous pouvez ignorer cette étape de validation et enregistrer votre politique de compartiment Amazon S3 à tout moment.

Pour prévisualiser l'accès externe à votre compartiment, vous devez disposer d'un analyseur de compte actif dans la région de votre compartiment avec le compte comme zone de confiance. Vous devez également avoir des autorisations requises pour l'utilisation de l'IAM Access Analyzer et sa prévisualisation Pour plus d'informations sur l'activation de l'IAM Access Analyzer et les autorisations requises, veuillez consulter Démarrage avec AWS Identity and Access Management Access Analyzer.

Pour prévisualiser l'accès à votre compartiment Amazon S3 lorsque vous créez ou modifiez votre politique de compartiment

1. Une fois la création ou la modification de votre politique de compartiment terminée, assurez-vous que votre politique est une politique de compartiment Amazon S3 valide. L'ARN de politique doit correspondre à l'ARN du compartiment, et les éléments de politique doivent être valides.
2. En dessous de la politique, sous Preview external access (Prévisualiser l'accès externe), sélectionnez un analyseur de comptes actif, puis sélectionnez Preview (Prévisualiser). Une prévisualisation des résultats d'IAM Access Analyzer est générée pour votre compartiment. La prévisualisation analyse la politique de compartiment Amazon S3 affichée, ainsi que les autorisations de compartiment existantes. Cela inclut les paramètres BPA du compartiment et du compte, l'ACL du compartiment, les points d'accès Amazon S3 et les points d'accès multi-région attachés au compartiment, ainsi que leurs politiques et paramètres BPA.
3. Une fois la prévisualisation de l'accès terminée, une prévisualisation des résultats d'IAM Access Analyzer s'affiche. Chaque résultat signale une instance d'un principal en dehors du compte

avec accès à votre compartiment après avoir enregistré la politique. Vous pouvez valider l'accès à votre compartiment en analysant chaque résultat. L'en-tête du résultat fournit un résumé de l'accès, et vous pouvez développer le résultat afin d'analyser les détails d'un résultat. La recherche de badges fournit un contexte sur la façon dont l'enregistrement de la politique de compartiment modifierait l'accès au compartiment. Par exemple, ils vous aident à vérifier si la modification de politique introduit de nouveaux résultats ou résout les résultats existants pour un accès externe :

   a. Nouveau : indique un résultat pour un nouvel accès externe que la politique introduirait.
   b. Résolu : indique un résultat pour un accès externe existant que la politique supprimerait.
   c. Archivé : indique un résultat pour un nouvel accès externe qui serait automatiquement archivé en fonction des règles d'archivage de l'analyseur qui définissent à quel moment les résultats doivent être marqués comme prévus.
   d. Existant : indique un résultat existant pour un accès externe qui resterait inchangé.
   e. Public : si un résultat concerne l'accès public à la ressource, il aura un badge Public en plus de l'un des badges ci-dessus.

4. Si vous identifiez un accès externe que vous n'envisagez pas d'introduire ou de supprimer, vous pouvez réviser la politique, puis choisir Preview (Prévisualiser) jusqu'à ce que vous réalisiez l'accès externe prévu. Si vous avez un résultat étiqueté Public, nous vous recommandons de réviser la politique afin de supprimer l'accès public avant de choisir Save changes (Enregistrer les modifications). La prévisualisation de l'accès est une étape facultative, et vous pouvez choisir Save changes (Enregistrer les modifications) à tout moment.

Prévisualisation de l'accès avec IAM Access Analyzer APIs

Vous pouvez utiliser IAM Access Analyzer APIs pour prévisualiser l'accès public et entre comptes à vos compartiments Amazon S3, à vos AWS KMS clés, à vos rôles IAM, à vos files d'attente Amazon SQS et à vos secrets Secrets Manager. Vous pouvez prévisualiser l'accès en fournissant des autorisations proposées pour une ressource existante que vous possédez ou une nouvelle ressource que vous voulez déployer.

Pour prévisualiser l'accès externe à votre ressource, vous devez disposer d'un analyseur de compte actif pour le compte et la région de la ressource. Vous devez également avoir des autorisations requises pour l'utilisation de l'IAM Access Analyzer et sa prévisualisation Pour plus d'informations sur l'activation de l'IAM Access Analyzer et les autorisations requises, veuillez consulter Démarrage avec AWS Identity and Access Management Access Analyzer.

Pour prévisualiser l'accès à une ressource, vous pouvez utiliser l'opération `CreateAccessPreview` et fournir l'ARN de l'analyseur et la configuration du contrôle d'accès pour la ressource. Le service renvoie l'ID unique pour la prévisualisation de l'accès, que vous pouvez utiliser pour en vérifier l'état avec l'opération `GetAccessPreview`. Lorsque le statut est `Completed`, vous pouvez utiliser l'opération `ListAccessPreviewFindings` pour récupérer les résultats générés pour la prévisualisation de l'accès. Les opérations `GetAccessPreview` et `ListAccessPreviewFindings` récupèrent les prévisualisations d'accès et les résultats créés dans un délai d'environ 24 heures.

Chaque résultat récupéré contient des détails du résultat, qui décrivent l'accès. Un statut de prévisualisation du résultat indiquant si le résultat serait `Active`, `Archived` ou `Resolved` après le déploiement des autorisations, et un `changeType`. Le `changeType` donne un contexte sur la comparaison du résultat de la prévisualisation de l'accès à celui existant et identifié dans l'IAM Access Analyzer :

- Nouveau : le résultat concerne l'accès nouvellement introduit.
- Inchangé : le résultat de la prévisualisation est un résultat existant qui resterait inchangé.
- Modifié : le résultat de la prévisualisation est un résultat existant dont le statut changerait.

Le `status` et le `changeType` vous aident à comprendre comment la configuration de la ressource modifierait l'accès à la ressource existante. Si le `changeType` est `Unchanged` ou Modifié, le résultat contiendra également l'ID et le statut existants du résultat dans l'IAM Access Analyzer. Par exemple, un résultat `Changed` avec le statut de prévisualisation `Resolved` et un statut existant `Active` indique que le résultat existant `Active` pour la ressource deviendrait `Resolved` à la suite de la modification proposée des autorisations.

Vous pouvez utiliser l'opération `ListAccessPreviews` pour récupérer une liste des prévisualisations d'accès pour l'analyseur spécifié. Cette opération récupère les informations sur la prévisualisation de l'accès créée dans un délai d'environ une heure.

En général, si la prévisualisation de l'accès concerne une ressource existante et que vous ne spécifiez pas d'option de configuration, la prévisualisation de l'accès utilisera la configuration de ressource existante par défaut. Si la prévisualisation de l'accès concerne une nouvelle ressource et que vous ne spécifiez pas d'option de configuration, la prévisualisation de l'accès utilisera la valeur par défaut en fonction du type de ressource. Pour connaître les cas de configuration pour chaque type de ressource, veuillez consulter ce qui suit.

Prévisualiser l'accès à votre compartiment Amazon S3

Pour créer un aperçu d'accès pour un nouveau compartiment Amazon S3 ou un compartiment Amazon S3 existant dont vous êtes propriétaire, vous pouvez proposer une configuration de compartiment en spécifiant la politique du compartiment Amazon S3, les paramètres BPA du compartiment et les points d'accès Amazon S3, y compris les points d'accès multirégionaux, attachés au compartiment. ACLs

> **Note**
>
> Avant de tenter de créer un aperçu d'accès pour un nouveau compartiment, nous vous recommandons d'appeler l'HeadBucketopération Amazon S3 pour vérifier si le compartiment nommé existe déjà. Cette opération est utile pour déterminer si un compartiment existe et si vous avez l'autorisation d'y accéder.

Politique de compartiment : si la configuration concerne un compartiment Amazon S3 existant et que vous ne spécifiez pas la politique de compartiment Amazon S3, la prévisualisation de l'accès utilise la politique existante attachée au compartiment. Si la prévisualisation de l'accès concerne une nouvelle ressource et que vous ne spécifiez pas la politique de compartiment Amazon S3, la prévisualisation de l'accès suppose un compartiment sans politique. Pour proposer la suppression d'une politique de compartiment existante, vous pouvez spécifier une chaîne vide. Pour plus d'informations sur les limites de la politique de compartiment prise en charge, veuillez consulter Bucket policy examples (Exemples de politique de compartiment).

Octrois d'ACL de compartiment : vous pouvez proposer jusqu'à 100 octrois d'ACL par compartiment. Si la configuration d'octroi proposée concerne un compartiment existant, la prévisualisation de l'accès utilise la liste proposée des configurations d'octroi à la place des autorisations existantes. Sinon, la prévisualisation de l'accès utilise les autorisations existantes pour le compartiment.

Points d'accès au compartiment : l'analyse prend en charge jusqu'à 100 points d'accès, dont des points d'accès multi-région, par compartiment, et vous pouvez proposer jusqu'à dix nouveaux points d'accès par compartiment. Si la configuration de point d'accès Amazon S3 proposée concerne un compartiment existant, la prévisualisation de l'accès utilise la configuration proposée de points d'accès à la place des points d'accès existants. Pour proposer un point d'accès sans politique, vous pouvez fournir une chaîne vide en tant que politique de point d'accès. Pour plus d'informations sur les limites de la politique de point d'accès, veuillez consulter Access points restrictions and limitations (Limites et restrictions des points d'accès).

Bloquer la configuration de l'accès public : si la configuration proposée concerne un compartiment Amazon S3 existant et que vous ne spécifiez pas la configuration, la prévisualisation de l'accès utilise le paramètre existant. Si la configuration proposée concerne un nouveau compartiment et que vous ne spécifiez pas la configuration BPA du compartiment, la prévisualisation de l'accès utilise `false`. Si la configuration proposée concerne un nouveau point d'accès ou point d'accès multi-région et que vous ne spécifiez pas la configuration BPA du point d'accès, la prévisualisation de l'accès utilise `true`.

Accès prévisualisé à votre AWS KMS clé

Pour créer un aperçu de l'accès pour une nouvelle AWS KMS clé ou une AWS KMS clé existante que vous possédez, vous pouvez proposer une configuration de AWS KMS clé en spécifiant la politique des clés et la configuration des AWS KMS autorisations.

AWS KMS politique de clé — Si la configuration concerne une clé existante et que vous ne spécifiez pas la politique de clé, l'aperçu de l'accès utilise la politique existante pour la clé. Si la prévisualisation de l'accès concerne une nouvelle ressource et que vous ne spécifiez pas la politique de clé, la prévisualisation de l'accès utilise la politique de clé par défaut. La politique de clé proposée ne peut pas être une chaîne vide.

AWS KMS subventions — L'analyse prend en charge jusqu'à 100 autorisations KMS par configuration*.* Si la configuration de subvention proposée concerne une clé existante, l'aperçu de l'accès utilise la liste proposée de configurations de subventions à la place des subventions existantes. Sinon, la prévisualisation de l'accès utilise les octrois existants pour la clé.

Prévisualiser l'accès à votre rôle IAM

Pour créer une prévisualisation de l'accès pour un nouveau rôle IAM ou un rôle IAM existant que vous possédez, vous pouvez proposer une configuration de rôle IAM en spécifiant la politique d'approbation.

Politique d'approbation de rôle : si la configuration concerne un nouveau rôle IAM, vous devez spécifier la politique d'approbation. Si la configuration concerne un rôle IAM existant que vous possédez et que vous ne proposez pas la politique d'approbation, la prévisualisation de l'accès utilise la politique d'approbation existante pour le rôle. La politique d'approbation proposée ne peut pas être une chaîne vide.

Prévisualiser l'accès à votre file d'attente Amazon SQS

Pour créer une prévisualisation de l'accès pour une nouvelle file d'attente Amazon SQS ou une file d'attente Amazon SQS existante que vous possédez, vous pouvez proposer une configuration de file d'attente Amazon SQS en spécifiant la politique Amazon SQS pour la file d'attente.

Politique de file d'attente Amazon SQS : si la configuration concerne une file d'attente Amazon SQS existante et que vous ne spécifiez pas la politique Amazon SQS, la prévisualisation de l'accès utilise la politique Amazon SQS existante pour la file d'attente. Si la prévisualisation de l'accès concerne une nouvelle ressource et que vous ne spécifiez pas la politique, la prévisualisation de l'accès suppose une file d'attente Amazon SQS sans politique. Pour proposer la suppression d'une politique de file d'attente Amazon SQS existante, vous pouvez spécifier une chaîne vide pour la politique Amazon SQS.

Prévisualiser l'accès à votre secret Secrets Manager

Pour créer un aperçu de l'accès à un nouveau secret du Gestionnaire de Secrets ou à un secret du Gestionnaire de Secrets existant dont vous êtes propriétaire, vous pouvez proposer une configuration du secret du Gestionnaire de secrets en spécifiant la politique secrète et la clé de AWS KMS chiffrement optionnelle.

Politique de secret : si la configuration concerne un secret existant et que vous ne spécifiez pas la politique secrète, la prévisualisation de l'accès utilise la politique existante pour le secret. Si la prévisualisation de l'accès concerne une nouvelle ressource et que vous ne spécifiez pas la politique, la prévisualisation de l'accès suppose un secret sans politique. Pour proposer la suppression d'une politique existante, vous pouvez spécifier une chaîne vide.

AWS KMS clé de chiffrement — Si la configuration proposée concerne un nouveau secret et que vous ne spécifiez pas l'ID de la AWS KMS clé, l'aperçu de l'accès utilise la clé KMS par défaut du AWS compte. Si vous spécifiez une chaîne vide pour l'ID de AWS KMS clé, l'aperçu de l'accès utilise la clé KMS par défaut du AWS compte.

Vérifications de validation des politiques

L'analyseur d'accès IAM fournit des contrôles de politique qui aident à valider vos politiques IAM avant de les attacher à une entité. Il s'agit notamment des vérifications de politique de base fournies par la validation des politiques pour valider votre politique par rapport à la grammaire des politiques et aux bonnes pratiques AWS. Vous pouvez afficher les résultats des vérifications de validation de

politique qui incluent des avertissements de sécurité, des erreurs, des avertissements généraux et des suggestions pour votre politique.

Vous pouvez utiliser des vérifications de politique personnalisées pour vérifier les nouveaux accès en fonction de vos normes de sécurité. Des frais sont associés à chaque vérification pour un nouvel accès. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

Comment fonctionnent les vérifications de politique personnalisées

Vous pouvez valider vos politiques par rapport aux normes de sécurité spécifiées à l'aide de vérifications de politique personnalisées AWS Identity and Access Management Access Analyzer . Les types de vérifications de politiques personnalisées que vous pouvez exécuter sont les suivants :

- Vérifier par rapport à une stratégie de référence : Lorsque vous modifiez une politique, vous pouvez vérifier si celle mise à jour accorde un nouvel accès par rapport à une de référence, telle qu'une version existante de cette politique. Vous pouvez exécuter cette vérification lorsque vous modifiez une politique à l'aide de AWS Command Line Interface (AWS CLI), de l'API (API) IAM Access Analyzer ou de l'éditeur de règles JSON dans la console IAM.

  > **Note**
  >
  > Les vérifications de politiques personnalisées de l'analyseur d'accès IAM autorisent l'ajout de caractères génériques dans l'élément `Principal` pour les politiques de ressources de référence.

- Vérifiez par rapport à une liste d'actions ou de ressources IAM : vous pouvez vérifier que des actions ou des ressources IAM spécifiques ne sont pas autorisées par votre politique. Si seules des actions sont spécifiées, l'analyseur d'accès IAM vérifie l'accès aux actions sur toutes les ressources de la politique. Si seules les ressources sont spécifiées, l'analyseur d'accès IAM vérifie quelles actions ont accès aux ressources spécifiées. Si des actions et des ressources sont spécifiées, l'analyseur d'accès IAM vérifie quelles sont les actions spécifiées qui ont accès aux ressources spécifiées. Vous pouvez exécuter cette vérification lorsque vous créez ou modifiez une politique à l'aide de l'API AWS CLI ou de l'API.
- Vérifier l'accès public : vous pouvez vérifier si une politique de ressources peut accorder un accès public à un type de ressource spécifique. Vous pouvez exécuter cette vérification lorsque vous créez ou modifiez une politique à l'aide de l'API AWS CLI ou de l'API. Ce type de vérification des politiques personnalisées diffère de la prévisualisation de l'accès, car la vérification ne nécessite aucun contexte de compte ou d'analyseur d'accès externe. Les prévisualisations de l'accès

vous permettent de prévisualiser les résultats de l'analyseur d'accès IAM avant de déployer les autorisations relatives aux ressources, tandis que la vérification personnalisée détermine si un accès public peut être accordé par une politique.

Des frais sont associés à chaque vérification de politique personnalisée. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

Vous pouvez exécuter des vérifications de politique personnalisées sur les politiques basées sur l'identité et les ressources. Les vérifications de politique personnalisées ne reposent pas sur des techniques de correspondance de modèles ou sur l'examen des journaux d'accès pour déterminer si un nouvel accès ou spécifique est autorisé par une politique. À l'instar des résultats des accès externes, les vérifications de politique personnalisées sont basées sur Zelkova. Zelkova traduit les politiques IAM en déclarations logiques équivalentes, et exécute une suite de solveurs logiques polyvalents et spécialisés (théories du module de satisfiabilité) contre le problème. Pour vérifier l'existence d'un accès nouveau ou spécifié, l'analyseur d'accès IAM applique Zelkova de manière répétée à une politique. Les requêtes deviennent de plus en plus spécifiques pour caractériser les classes de comportements que la politique autorise sur la base du contenu de la politique. Pour en savoir plus sur les théories du module de satisfiabilité, consultez Théories du module de satisfiabilité.

Dans de rares cas, l'analyseur d'accès IAM n'est pas en mesure de déterminer complètement si une déclaration de politique octroie un accès nouveau ou spécifique. Dans ces cas, il commet une erreur en déclarant un faux positif en échouant à la vérification de politique personnalisées. L'analyseur d'accès IAM est conçu pour fournir une évaluation exhaustive des politiques et s'efforce de réduire les faux négatifs. Cette approche signifie que l'analyseur d'accès IAM offre un degré élevé d'assurance qu'une vérification réussie signifie que l'accès n'a pas été accordé par la politique. Vous pouvez inspecter les vérifications qui ont échoué manuellement en consultant la déclaration de politique indiquée dans la réponse de l'analyseur d'accès IAM.

Exemples de politiques de référence de vérification des nouveaux accès

Vous pouvez trouver des exemples de politiques de référence et apprendre à configurer et exécuter une vérification de stratégie personnalisée pour les nouveaux accès dans le référentiel d'exemples de contrôles de politiques personnalisés d'IAM Access Analyzer sur. GitHub

Avant d'utiliser ces exemples

Avant d'utiliser ces exemples de politiques de référence, effectuez les opérations suivantes :

- Vérifiez attentivement et personnalisez les politiques de référence en fonction de vos exigences uniques.
- Testez soigneusement les politiques de référence dans votre environnement avec les Services AWS que vous utilisez.

  Les politiques de référence illustrent la mise en œuvre et l'utilisation de vérifications de politique personnalisées. Ils ne sont pas destinés à être interprétés comme des recommandations ou des bonnes pratiques AWS officielles à mettre en œuvre exactement comme indiqué. Il est de votre responsabilité de tester soigneusement les politques de référence afin de déterminer si elles conviennent pour répondre aux exigences de sécurité de votre environnement.
- Les vérifications de politique personnalisées sont indépendantes de l'environnement dans leur analyse. Leur analyse ne prend en compte que les informations contenues dans les politiques d'entrée. Par exemple, les contrôles de politique personnalisés ne permettent pas de vérifier si un compte appartient à une AWS organisation spécifique. Par conséquent, les vérifications de politique personnalisées ne peuvent pas comparer les nouveaux accès en fonction des valeurs des clés de condition pour les clés de condition `aws:PrincipalOrgId` et `aws:PrincipalAccount`.

Inspection des vérifications de politiques personnalisées en échec

Lorsqu'une vérification de politique personnalisée échoue, la réponse de l'analyseur d'accès IAM inclut l'ID d'instruction (`Sid`) de celle de politique responsable de l'échec de la vérification. Bien que l'ID de déclaration soit un élément de politique facultatif, nous vous recommandons d'ajouter un ID de déclaration pour chaque déclaration de politique. La vérification de politique personnalisée renvoie également un index de déclaration pour aider à identifier la raison de l'échec de la vérification. L'index de déclaration suit une numérotation à base de zéro, où la première déclaration est référencée par 0. Lorsque plusieurs déclarations entraînent l'échec d'une vérification, celle-ci ne renvoie qu'un seul ID de déclaration à la fois. Nous vous recommandons de corriger la déclaration mise en évidence dans le motif et de réexécuter la vérification jusqu'à ce qu'elle soit validée.

Validation des politiques à l'aide de l'analyseur d'accès IAM

Vous pouvez valider vos politiques à l'aide de AWS Identity and Access Management Access Analyzer la validation des politiques. Vous pouvez créer ou modifier une politique à l'aide de l' AWS CLI AWS API ou de l'éditeur de stratégie JSON dans la console IAM. L'analyseur d'accès

IAM valide votre politique par rapport à la grammaire de politique IAM et aux bonnes pratiques AWS. Vous pouvez afficher les résultats des vérifications de validation de politique qui incluent des avertissements de sécurité, des erreurs, des avertissements généraux et des suggestions pour votre politique. Ces résultats fournissent des recommandations exploitables qui vous aident à créer des politiques fonctionnelles et conformes aux bonnes pratiques en matière de sécurité. Pour afficher la liste des vérifications des politiques de base exécutées par l'analyseur d'accès IAM, consultez Référence de vérification de validation de politique IAM.

Validation des politiques dans IAM (console)

Vous pouvez afficher les résultats générés par la validation de politiques de l'analyseur d'accès IAM lorsque vous créez ou modifiez une politique gérée dans la console IAM. Vous pouvez également afficher ces résultats pour les politiques d'utilisateur ou de rôle en ligne. L'analyseur d'accès IAM ne génère pas ces résultats pour les politiques de groupe en ligne.

Pour afficher les résultats générés par les vérifications de politique pour les politiques IAM JSON

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Commencez à créer ou à modifier une politique selon l'une des méthodes suivantes :
   a. Pour créer une politique gérée, accédez à la page Policies (Politiques) et créez une politique. Pour de plus amples informations, veuillez consulter Création de politiques à l'aide de l'éditeur JSON.
   b. Pour afficher les vérifications de politique d'une politique gérée par le client existante, accédez à la page Politiques, sélectionnez le nom d'une politique, puis sélectionnez Modifier. Pour de plus amples informations, veuillez consulter Modification de politiques gérées par le client (console).
   c. Pour afficher les vérifications de politique d'une politique en ligne sur un utilisateur ou un rôle, accédez à la page Utilisateurs ou Rôles, sélectionnez le nom d'un utilisateur ou d'un rôle, sélectionnez le nom de la politique sous l'onglet Autorisations, puis choisissez Modifier. Pour de plus amples informations, veuillez consulter Modification de politiques en ligne (console).
3. Dans l'éditeur de politique, sélectionnez l'onglet JSON.
4. Dans le panneau de validation de la politique, sous la politique, sélectionnez un ou plusieurs des onglets suivants. Les noms des onglets indiquent également le nombre de chaque type de résultat pour votre politique.

- Sécurité : consultez les avertissements si votre politique autorise un accès AWS considéré comme présentant un risque de sécurité en raison d'un accès trop permissif.
- Erreurs : affiche des erreurs si votre politique inclut des lignes qui empêchent la politique de fonctionner.
- Avertissements : affiche des avertissements si votre politique n'est pas conforme aux bonnes pratiques, mais que les problèmes ne constituent pas des risques de sécurité.
- Suggestions : affiche des suggestions si AWS recommande des améliorations qui n'affectent pas les autorisations de la politique.

5. Examinez les détails des résultats fournis par la vérification de la politique de l'IAM Access Analyzer. Chaque résultat indique l'emplacement du problème signalé. Pour en savoir plus sur les causes du problème et la façon de le résoudre, sélectionnez le lien Learn more (En savoir plus) en regard du résultat. Vous pouvez également rechercher la vérification de politique associée à chaque résultat sur la page de référence Access Analyzer policy checks (Vérifications de politique Access Analyzer).
6. Facultatif. Si vous modifiez une politique existante, vous pouvez exécuter une vérification de politique personnalisée pour déterminer si votre politique mise à jour accorde un nouvel accès par rapport à la version existante. Dans le panneau de validation de la politique ci-dessous, sélectionnez l'onglet Vérifier un nouvel accès, puis sélectionnez Vérifier la politique. Si les autorisations modifiées accordent un nouvel accès, la déclaration sera mise en évidence dans le volet de validation de la politique. Si vous n'avez pas l'intention d'accorder un nouvel accès, mettez à jour la déclaration de politique et choisissez Vérifier la politique jusqu'à ce qu'aucun nouvel accès ne soit détecté. Pour de plus amples informations, veuillez consulter Validation des politiques à l'aide des vérifications des politiques personnalisées de l'analyseur d'accès IAM.

> **Note**
>
> Des frais sont associés à chaque vérification pour un nouvel accès. Pour plus d'informations sur la tarification, consultez Tarification de l'analyseur d'accès IAM.

7. Mettez à jour votre politique pour résoudre les résultats.

> **Important**
>
> Testez soigneusement les politiques nouvelles ou modifiées avant de les implémenter dans votre flux de production.

8. Lorsque vous avez terminé, choisissez Next. Le Validateur de politique signale les erreurs de syntaxe qui ne sont pas signalées par l'analyseur d'accès IAM.

   > **Note**
   >
   > Vous pouvez basculer à tout moment entre les onglets Visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'onglet Visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour de plus amples informations, veuillez consulter Restructuration de politique.

9. Pour de nouvelles politiques, sur la page Examiner et créer, tapez un Nom de politique et une Description (facultative) pour la politique que vous créez. Examinez les Autorisations définies dans cette politique pour voir celles accordées par votre politique. Sélectionnez ensuite Créer une politique pour enregistrer votre travail.

   Pour les politiques existantes, à la page Examiner et enregistrer, passez en revue les Autorisations définies dans cette politique pour voir celles qui sont accordées par votre politique. Choisissez l'option Définir cette nouvelle version comme version par défaut pour enregistrer la version mise à jour comme version par défaut de la politique. Choisissez ensuite Enregistrer les modifications pour enregister votre travail.

Validation des politiques à l'aide d'IAM Access Analyzer (ou API)AWS CLI AWS

Vous pouvez afficher les résultats générés par la validation de politique de l'analyseur d'accès IAM à partir d' AWS Command Line Interface (AWS CLI).

Pour consulter les résultats générés par la validation des politiques (AWS CLI ou AWS API) d'IAM Access Analyzer

Utilisez l'une des options suivantes :

- AWS CLI : aws accessanalyzer validate-policy
- AWS API : ValidatePolicy

Référence de vérification de validation de politique IAM

Vous pouvez valider vos politiques à l'aide de AWS Identity and Access Management Access Analyzer la validation des politiques. Vous pouvez créer ou modifier une politique à l'aide de l' AWS

CLI AWS API ou de l'éditeur de règles JSON de la console IAM. L'analyseur d'accès IAM valide votre politique par rapport à la grammaire de politique IAM et aux bonnes pratiques AWS. Vous pouvez afficher les résultats des vérifications de validation de politique qui incluent des avertissements de sécurité, des erreurs, des avertissements généraux et des suggestions pour votre politique. Ces résultats fournissent des recommandations exploitables qui vous aident à créer des politiques fonctionnelles et conformes aux bonnes pratiques en matière de sécurité. La liste des vérifications de politiques de base fournies par l'analyseur d'accès IAM est présentée ci-dessous. Pas de frais supplémentaires pour l'exécution des vérifications de validation de la politique. Pour en savoir plus sur la validation des politiques à l'aide de la validation des politiques, consultez Validation des politiques à l'aide de l'analyseur d'accès IAM.

Erreur : compte d'ARN non autorisé

Code d'erreur : ARN_ACCOUNT_NOT_ALLOWED

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
ARN account not allowed: The service {{service}} does not support specifying an account
 ID in the resource ARN.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The service {{service}} does not support specifying an account ID in
 the resource ARN."
```

Résolution de l'erreur

Supprimez l'ID de compte de l'ARN de ressource. Les ARN de ressources de certains AWS services ne permettent pas de spécifier un identifiant de compte.

Par exemple, Amazon S3 ne prend pas en charge un ID de compte en tant qu'espace de noms dans les ARN de compartiment. Le nom d'un compartiment Amazon S3 est unique au monde et l'espace de noms est partagé par tous les AWS comptes. Pour afficher tous les types de ressources disponibles dans Amazon S3, veuillez consulter Types de ressources définis par Amazon S3 dans la Référence des autorisations de service.

Termes connexes

- Ressources de politique
- Identificateurs de compte
- ARN de la ressource
- AWS ressources de service avec formats ARN

Erreur : région d'ARN non autorisée

Code d'erreur : ARN_REGION_NOT_ALLOWED

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
ARN Region not allowed: The service {{service}} does not support specifying a Region in
 the resource ARN.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The service {{service}} does not support specifying a Region in the
 resource ARN."
```

Résolution de l'erreur

Supprimez la région de l'ARN de ressource. Les ARN de ressources de certains AWS services ne permettent pas de spécifier une région.

Par exemple, IAM est un service global. La partie Région d'un ARN de ressource IAM est toujours vide. Les ressources IAM sont mondiales, comme l'est un AWS compte aujourd'hui. Par exemple, une fois connecté en tant qu'utilisateur IAM, vous pouvez accéder à des AWS services dans n'importe quelle région géographique.

- Ressources de politique
- ARN de la ressource
- AWS ressources de service avec formats ARN

Erreur : non-correspondance du type de données

Code d'erreur : DATA_TYPE_MISMATCH

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Data type mismatch: The text does not match the expected JSON data type {{data_type}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The text does not match the expected JSON data type {{data_type}}."
```

Résolution de l'erreur

Mettez à jour le texte pour utiliser le type de données pris en charge.

Par exemple, la clé de condition globale `Version` a besoin d'un type de données `String`. Si vous fournissez une date ou un entier, le type de données ne correspondra pas.

Termes connexes

- Clés de condition globale
- Éléments de politique JSON IAM : Opérateurs de condition

Erreur : clés dupliquées avec une casse différente

Code d'erreur : DUPLICATE_KEYS_WITH_DIFFERENT_CASE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Duplicate keys with different case: The condition key {{key}} appears more than once
 with different capitalization in the same condition block. Remove the duplicate
 condition keys.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key {{key}} appears more than once with different
 capitalization in the same condition block. Remove the duplicate condition keys."
```

Résolution de l'erreur

Examinez les clés de condition similaires dans le même bloc de condition et utilisez la même casse pour toutes les instances.

On appelle bloc de condition le texte inclus dans l'élément `Condition`d'une instruction de politique. Les noms de clé de condition ne sont pas sensibles à la casse. La sensibilité à la casse des valeurs des clés de condition dépend de l'opérateur de condition que vous utilisez. Pour de plus amples informations sur la sensibilité à la casse des clés de condition, veuillez consulter Éléments de la politique JSON IAM : Condition.

Termes connexes

- Conditions
- Bloc de condition
- Clés de condition globale
- AWS clés d'état de service

Erreur : action non valide

Code d'erreur : INVALID_ACTION

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid action: The action {{action}} does not exist. Did you mean {{valid_action}}?
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The action {{action}} does not exist. Did you mean
 {{valid_action}}?"
```

Résolution de l'erreur

L'action que vous avez spécifiée n'est pas valide. Cela peut se produire si vous saisissez de façon incorrecte le préfixe du service ou le nom de l'action. Pour certains problèmes courants, la vérification de politique renvoie une action suggérée.

Termes connexes

- Actions de politique
- AWS actions de service

AWS politiques gérées avec cette erreur

AWS les politiques gérées vous permettent de commencer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Les politiques AWS gérées suivantes incluent des actions non valides dans leurs déclarations de politique. Les actions non valides n'affectent pas les autorisations octroyées par la politique. Lorsque vous utilisez une stratégie AWS gérée comme référence pour créer votre stratégie gérée, vous AWS recommande de supprimer les actions non valides de votre politique.

- AmazonEMRFullAccessPolicy_v2
- CloudWatchSyntheticsFullAccess

Erreur : compte d'ARN non valide

Code d'erreur : INVALID_ARN_ACCOUNT

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid ARN account: The resource ARN account ID {{account}} is not valid. Provide a
 12-digit account ID.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The resource ARN account ID {{account}} is not valid. Provide a 12-digit account ID."
```

Résolution de l'erreur

Mettez à jour l'ID de compte dans l'ARN de ressource. Les ID de compte sont des nombres entiers à 12 chiffres. Pour savoir comment consulter votre identifiant de compte, voir Trouver votre identifiant de AWS compte.

Termes connexes

- Ressources de politique
- Identificateurs de compte
- ARN de la ressource
- AWS ressources de service avec formats ARN

Erreur : préfixe d'ARN non valide

Code d'erreur : INVALID_ARN_PREFIX

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid ARN prefix: Add the required prefix (arn) to the resource ARN.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add the required prefix (arn) to the resource ARN."
```

Résolution de l'erreur

AWS les ARN de ressources doivent inclure le `arn:` préfixe requis.

Termes connexes

- Ressources de politique
- ARN de la ressource
- AWS ressources de service avec formats ARN

Erreur : région d'ARN non valide

Code d'erreur : INVALID_ARN_REGION

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid ARN Region: The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region."
```

Résolution de l'erreur

Le type de ressource n'est pas pris en charge dans la région spécifiée. Pour un tableau des AWS services pris en charge dans chaque région, consultez le tableau des régions.

Termes connexes

- Ressources de politique
- ARN de la ressource
- Noms et codes des régions

Erreur : ARN de ressource non valide

Code d'erreur : INVALID_ARN_RESOURCE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid ARN resource: Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN."
```

Résolution de l'erreur

L'ARN de ressource doit correspondre aux spécifications des types de ressources connus. Pour afficher le format ARN attendu pour un service, voir Actions, ressources et clés de condition pour les AWS services. Choisissez le nom du service pour afficher ses types de ressources et formats d'ARN.

Termes connexes

- Ressources de politique
- ARN de la ressource
- AWS ressources de service avec formats ARN

Erreur : cas de service ARN non valide

Code d'erreur : INVALID_ARN_SERVICE_CASE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid ARN service case: Update the service name {{service}} in the resource ARN to
 use all lowercase letters.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Update the service name {{service}} in the resource ARN to use all
 lowercase letters."
```

Résolution de l'erreur

Le service dans l'ARN de ressource doit correspondre aux spécifications (notamment la casse) des préfixes de service. Pour afficher le préfixe d'un service, voir Actions, ressources et clés de condition pour les AWS services. Choisissez le nom du service et localisez son préfixe dans la première phrase.

Termes connexes

- Ressources de politique
- ARN de la ressource
- AWS ressources de service avec formats ARN

Erreur : type de données de condition non valide

Code d'erreur : INVALID_CONDITION_DATA_TYPE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid condition data type: The condition value data types do not match. Use condition
 values of the same JSON data type.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition value data types do not match. Use condition values of
 the same JSON data type."
```

Résolution de l'erreur

La valeur de la paire clé-valeur de condition doit correspondre au type de données de la clé de condition et de l'opérateur de condition. Pour afficher le type de données de clé de condition pour un service, voir Actions, ressources et clés de condition pour les AWS services. Choisissez le nom du service pour afficher les clés de condition de ce service.

Par exemple, la clé de condition globale `CurrentTime` prend en charge l'opérateur de condition `Date`. Si vous fournissez une chaîne ou un entier pour la valeur dans le bloc de condition, le type de données ne correspondra pas.

Termes connexes

- Conditions
- Bloc de condition
- Éléments de politique JSON IAM : Opérateurs de condition
- Clés de condition globale
- AWS clés d'état de service

Erreur : format de clé de condition non valide

Code d'erreur : INVALID_CONDITION_KEY_FORMAT

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid condition key format: The condition key format is not valid. Use the format
 service:keyname.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key format is not valid. Use the format
 service:keyname."
```

Résolution de l'erreur

La clé de la paire clé-valeur de condition doit correspondre aux spécifications du service. Pour afficher les clés de condition pour un service, voir Actions, ressources et clés de condition pour les AWS services. Choisissez le nom du service pour afficher les clés de condition de ce service.

Termes connexes

- Conditions
- Clés de condition globale
- AWS clés d'état de service

Erreur : valeurs booléennes multiples de condition non valides

Code d'erreur : INVALID_CONDITION_MULTIPLE_BOOLEAN

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid condition multiple Boolean: The condition key does not support multiple Boolean
 values. Use a single Boolean value.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key does not support multiple Boolean values. Use a
 single Boolean value."
```

Résolution de l'erreur

La clé de la paire clé-valeur de condition attend une valeur booléenne unique. Lorsque vous fournissez plusieurs valeurs booléennes, la correspondance de condition peut ne pas renvoyer les résultats attendus.

Pour afficher les clés de condition pour un service, voir Actions, ressources et clés de condition pour les AWS services. Choisissez le nom du service pour afficher les clés de condition de ce service.

- Conditions
- Clés de condition globale

- AWS clés d'état de service

Erreur : opérateur de condition non valide

Code d'erreur : INVALID_CONDITION_OPERATOR

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid condition operator: The condition operator {{operator}} is not valid. Use a
 valid condition operator.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition operator {{operator}} is not valid. Use a valid
 condition operator."
```

Résolution de l'erreur

Mettez à jour la condition pour utiliser un opérateur de condition pris en charge.

Termes connexes

- Éléments de politique JSON IAM : Opérateurs de condition
- Élément de condition
- Présentation des politiques JSON

Erreur : effet non valide

Code d'erreur : INVALID_EFFECT

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid effect: The effect {{effect}} is not valid. Use Allow or Deny.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The effect {{effect}} is not valid. Use Allow or Deny."
```

Résolution de l'erreur

Mettez à jour l'élément `Effect` pour utiliser un effet valide. Les valeurs valides pour `Effect` sont **`Allow`** et **`Deny`**.

Termes connexes

- Élément Effet
- Présentation des politiques JSON

Erreur : clé de condition globale non valide

Code d'erreur : INVALID_GLOBAL_CONDITION_KEY

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid global condition key: The condition key {{key}} does not exist. Use a valid
 condition key.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key {{key}} does not exist. Use a valid condition
 key."
```

Résolution de l'erreur

Mettez à jour la clé de condition dans la paire clé-valeur de condition pour utiliser une clé de condition globale prise en charge.

Les clés de condition globales sont des clés de condition avec un `aws:` préfixe. AWS les services peuvent prendre en charge les clés de condition globales ou fournir des clés spécifiques au service qui incluent leur préfixe de service. Par exemple, les clés de condition IAM incluent le préfixe `iam:`. Pour plus d'informations, consultez la section Actions, ressources et clés de condition pour les AWS services et choisissez le service dont vous souhaitez afficher les clés.

Termes connexes

- Clés de condition globale

Erreur : partition non valide

Code d'erreur : INVALID_PARTITION

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid partition: The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}"
```

Résolution de l'erreur

Mettez à jour l'ARN de ressource de sorte à inclure une partition prise en charge. Si vous avez inclus une partition prise en charge, le service ou la ressource peut ne pas prendre en charge la partition que vous avez incluse.

Une partition est un groupe de AWS régions. Chaque AWS compte est limité à une partition. Dans les régions classiques, utilisez la partition `aws`. Dans les régions de Chine, utilisez `aws-cn`.

Termes connexes

- Amazon Resource Names (ARN) : partitions

Erreur : élément de politique non valide

Code d'erreur : INVALID_POLICY_ELEMENT

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid policy element: The policy element {{element}} is not valid.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The policy element {{element}} is not valid."
```

Résolution de l'erreur

Mettez à jour la politique de sorte à inclure uniquement les éléments de politique JSON pris en charge.

Termes connexes

- Éléments de politique JSON

Erreur : format de principal non valide

Code d'erreur : INVALID_PRINCIPAL_FORMAT

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid principal format: The Principal element contents are not valid. Specify a key-
value pair in the Principal element.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The Principal element contents are not valid. Specify a key-value
 pair in the Principal element."
```

Résolution de l'erreur

Mettez à jour le format de principal de sorte à utiliser un format de paire clé-valeur pris en charge.

Vous pouvez spécifier un principal dans une politique basée sur la ressource, mais pas dans une politique basée sur l'identité.

Par exemple, pour définir l'accès pour tous les utilisateurs d'un AWS compte, utilisez le principal suivant dans votre politique :

```
"Principal": { "AWS": "123456789012" }
```

Termes connexes

- Éléments de politique JSON : principal
- Identity-based politiques et politiques fondées sur les ressources

Erreur : clé principale non valide

Code d'erreur : INVALID_PRINCIPAL_KEY

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid principal key: The principal key {{principal-key}} is not valid.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The principal key {{principal-key}} is not valid."
```

Résolution de l'erreur

Mettez à jour la clé dans la paire clé-valeur principale pour utiliser une clé principale prise en charge. Les clés principales suivantes sont prises en charge :

- AWS
- CanonicalUser
- Fédérée
- Service

Termes connexes

- Elément principal

Erreur : région non valide

Code d'erreur : INVALID_REGION

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid Region: The Region {{region}} is not valid. Update the condition value to a
 supported Region.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The Region {{region}} is not valid. Update the condition value to a
 supported Region."
```

Résolution de l'erreur

Mettez à jour la valeur de la paire clé-valeur de condition pour inclure une région prise en charge. Pour un tableau des AWS services pris en charge dans chaque région, consultez le tableau des régions.

Termes connexes

- Ressources de politique

- ARN de la ressource
- Noms et codes des régions

Erreur : service non valide

Code d'erreur : INVALID_SERVICE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid service: The service {{service}} does not exist. Use a valid service name.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The service {{service}} does not exist. Use a valid service name."
```

Résolution de l'erreur

Le préfixe de service dans la clé d'action ou de condition doit correspondre aux spécifications (notamment la casse) des préfixes de service. Pour afficher le préfixe d'un service, voir Actions, ressources et clés de condition pour les AWS services. Choisissez le nom du service et localisez son préfixe dans la première phrase.

Termes connexes

- Services connus et leurs actions, ressources et clés de condition

Erreur : clé de condition de service non valide

Code d'erreur : INVALID_SERVICE_CONDITION_KEY

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid service condition key: The condition key {{key}} does not exist in the service
 {{service}}. Use a valid condition key.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key {{key}} does not exist in the service {{service}}.
 Use a valid condition key."
```

Résolution de l'erreur

Mettez à jour la clé dans la paire clé-valeur de condition pour utiliser une clé de condition connue pour le service. Les noms de clé de condition globale commencent par le préfixe `aws`. Les services AWS peuvent fournir des clés spécifiques au service qui possèdent le préfixe du service. Pour afficher le préfixe d'un service, voir Actions, ressources et clés de condition pour les AWS services.

Termes connexes

- Clés de condition globale
- Services connus et leurs actions, ressources et clés de condition

Erreur : service non valide en action

Code d'erreur : INVALID_SERVICE_IN_ACTION

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid service in action: The service {{service}} specified in the action does not
 exist. Did you mean {{service2}}?
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The service {{service}} specified in the action does not exist. Did
 you mean {{service2}}?"
```

Résolution de l'erreur

Le préfixe de service dans l'action doit correspondre aux spécifications (notamment la casse) des préfixes de service. Pour afficher le préfixe d'un service, voir Actions, ressources et clés de condition pour les AWS services. Choisissez le nom du service et localisez son préfixe dans la première phrase.

Termes connexes

- Élément Action
- Services connus et leurs actions

Erreur : variable non valide pour l'opérateur

Code d'erreur : INVALID_VARIABLE_FOR_OPERATOR

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid variable for operator: Policy variables can only be used with String and ARN
 operators.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Policy variables can only be used with String and ARN operators."
```

Résolution de l'erreur

Vous pouvez utiliser des variables de politique dans l'élément `Resource` et les comparaisons de chaîne de l'élément `Condition`. Les conditions prennent en charge des variables lorsque vous utilisez des opérateurs chaîne ou des opérateurs ARN. Les opérateurs de chaîne incluent `StringEquals`, `StringLike` et `StringNotLike`. Les opérateurs ARN incluent `ArnEquals` et `ArnLike`. Vous ne pouvez pas utiliser une variable de politique avec d'autres opérateurs, tels que les opérateurs Numeric, Date, Boolean, Binary, IP Address ou Null operators.

Termes connexes

- Utilisation de variables de politique dans l'élément Condition
- Élément de condition

Erreur : version non valide

Code d'erreur : INVALID_VERSION

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid version: The version {{version}} is not valid. Use one of the following
 versions: {{versions}}
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The version {{version}} is not valid. Use one of the following
 versions: {{versions}}"
```

Résolution de l'erreur

L'élément `Version` de politique spécifie les règles de syntaxe du langage AWS utilisées pour traiter une politique. Pour utiliser toutes les fonctions de politique disponibles, incluez l'élément `Version` suivant avant l'élément `Statement` dans l'ensemble de vos politiques.

```
"Version": "2012-10-17"
```

Termes connexes

- Élément Version

Erreur : erreur de syntaxe Json

Code d'erreur : JSON_SYNTAX_ERROR

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Json syntax error: Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}."
```

Résolution de l'erreur

Votre politique inclut une erreur de syntaxe. Vérifiez votre syntaxe JSON.

Termes connexes

- Validateur JSON
- Référence des éléments de politique JSON IAM
- Présentation des politiques JSON

Erreur : erreur de syntaxe Json

Code d'erreur : JSON_SYNTAX_ERROR

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Json syntax error: Fix the JSON syntax error.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Fix the JSON syntax error."
```

Résolution de l'erreur

Votre politique inclut une erreur de syntaxe. Vérifiez votre syntaxe JSON.

Termes connexes

- Validateur JSON
- Référence des éléments de politique JSON IAM
- Présentation des politiques JSON

Erreur : action manquante

Code d'erreur : MISSING_ACTION

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing action: Add an Action or NotAction element to the policy statement.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add an Action or NotAction element to the policy statement."
```

Résolution de l'erreur

AWS Les politiques JSON doivent inclure un `NotAction` élément `Action` ou.

Termes connexes

- Élément Action
- NotAction élément
- Présentation des politiques JSON

Erreur : champ ARN manquant

Code d'erreur : MISSING_ARN_FIELD

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing ARN field: Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource"
```

Résolution de l'erreur

Tous les champs de l'ARN de ressource doivent correspondre aux spécifications d'un type de ressource connu. Pour afficher le format ARN attendu pour un service, consultez la section Actions, ressources et clés de condition pour les AWS services. Choisissez le nom du service pour afficher ses types de ressources et formats d'ARN.

Termes connexes

- Ressources de politique
- ARN de la ressource
- AWS ressources de service avec formats ARN

Erreur : région ARN manquante

Code d'erreur : MISSING_ARN_REGION

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing ARN Region: Add a Region to the {{service}} resource ARN.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a Region to the {{service}} resource ARN."
```

Résolution de l'erreur

Les ARN de ressources pour la plupart des AWS services nécessitent que vous indiquiez une région. Pour un tableau des AWS services pris en charge dans chaque région, consultez le tableau des régions.

Termes connexes

- Ressources de politique
- ARN de la ressource
- Noms et codes des régions

Erreur : effet manquant

Code d'erreur : MISSING_EFFECT

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing effect: Add an Effect element to the policy statement with a value of Allow or
 Deny.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add an Effect element to the policy statement with a value of Allow
 or Deny."
```

Résolution de l'erreur

AWS Les politiques JSON doivent inclure un `Effect` élément dont la valeur est **`Allow`** et**`Deny`**.

Termes connexes

- Élément Effet
- Présentation des politiques JSON

Erreur : principal manquant

Code d'erreur : MISSING_PRINCIPAL

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing principal: Add a Principal element to the policy statement.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a Principal element to the policy statement."
```

Résolution de l'erreur

Resource-based les politiques doivent inclure un `Principal` élément.

Par exemple, pour définir l'accès pour tous les utilisateurs d'un AWS compte, utilisez le principal suivant dans votre politique :

```
"Principal": { "AWS": "123456789012" }
```

Termes connexes

- Elément principal
- Identity-based politiques et politiques fondées sur les ressources

Erreur : qualificateur manquant

Code d'erreur : MISSING_QUALIFIER

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing qualifier: The request context key {{key}} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The request context key {{key}} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy."
```

Résolution de l'erreur

Dans l'élément `Condition`, vous créez des expressions dans lesquelles vous utilisez des opérateurs de condition tels que égal ou inférieur à pour comparer une condition de la politique avec des clés et des valeurs dans le contexte de la demande. Pour les demandes qui incluent plusieurs valeurs pour une seule clé de condition, vous devez placer les conditions entre crochets comme un tableau ("Key2":["Value2A", "Value2B"]). Vous devez également utiliser les opérateurs définis `ForAllValues` ou `ForAnyValue` avec l'opérateur de condition `StringLike`. Ces qualificatifs ajoutent une fonctionnalité d'opération d'ensemble à l'opérateur de condition afin que vous puissiez tester plusieurs valeurs de demande par rapport à plusieurs valeurs de condition.

Termes connexes

- Clés de contexte à valeurs multiples
- Élément de condition

AWS politiques gérées avec cette erreur

AWS les politiques gérées vous permettent de commencer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Les politiques AWS gérées suivantes incluent un qualificatif manquant pour les clés de condition dans leurs déclarations de politique. Lorsque vous utilisez la politique AWS gérée comme référence pour créer votre politique gérée par le client, il vous AWS recommande d'ajouter les qualificatifs de clé de `ForAnyValue` condition `ForAllValues` ou à votre `Condition` élément.

- AWSGlueConsoleSageMakerNotebookFullAccess

Erreur : ressource manquante

Code d'erreur : MISSING_RESOURCE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing resource: Add a Resource or NotResource element to the policy statement.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a Resource or NotResource element to the policy statement."
```

Résolution de l'erreur

Toutes les politiques, à l'exception des politiques de confiance dans les rôles, doivent inclure un élément `Resource` ou `NotResource`.

Termes connexes

- Élément de ressource
- NotResource élément
- Identity-based politiques et politiques fondées sur les ressources
- Présentation des politiques JSON

Erreur : instruction manquante

Code d'erreur : MISSING_STATEMENT

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing statement: Add a statement to the policy
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a statement to the policy"
```

Résolution de l'erreur

Une politique JSON doit inclure une instruction.

Termes connexes

- Éléments de politique JSON

Erreur : Null avec Si existe

Code d'erreur : NULL_WITH_IF_EXISTS

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Null with if exists: The Null condition operator cannot be used with the IfExists
 suffix. Update the operator or the suffix.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The Null condition operator cannot be used with the IfExists suffix.
 Update the operator or the suffix."
```

Résolution de l'erreur

Vous pouvez ajouter `IfExists` à la fin de tous les noms d'opérateurs de condition, à l'exception de l'opérateur de condition `Null`. Utilisez un opérateur de condition `Null` pour vérifier si une clé de condition est présente lors de l'autorisation. Utilisez `...ifExists` pour spécifier que « If the policy key is present in the context of the request, process the key as specified in the policy. Si la clé n'est pas présente, évaluez l'élément de condition comme vrai. »

Termes connexes

- ... IfExists opérateurs conditionnels
- Opérateur de condition null
- Élément de condition

Erreur : erreur de syntaxe SCP, caractère générique Action

Code d'erreur : SCP_SYNTAX_ERROR_ACTION_WILDCARD

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
SCP syntax error action wildcard: SCP actions can include wildcards (*) only at the end
 of a string. Update {{action}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "SCP actions can include wildcards (*) only at the end of a string.
 Update {{action}}."
```

Résolution de l'erreur

AWS Organizations les politiques de contrôle des services (SCP) permettent de spécifier des valeurs dans les `NotAction` éléments `Action` ou. Cependant, ces valeurs peuvent inclure des caractères génériques (*) uniquement à la fin de la chaîne. Cela signifie que vous pouvez spécifier `iam:Get*` mais pas `iam:*role`.

Pour spécifier plusieurs actions, il est AWS recommandé de les répertorier individuellement.

Termes connexes

- Action et NotAction éléments du SCP
- Évaluation du SCP
- AWS Organizations politiques de contrôle des services
- Éléments de politique JSON IAM : Action

Erreur : erreur de syntaxe SCP, principal

Code d'erreur : SCP_SYNTAX_ERROR_PRINCIPAL

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
SCP syntax error principal: SCPs do not support specifying principals. Remove the
 Principal or NotPrincipal element.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "SCPs do not support specifying principals. Remove the Principal or
 NotPrincipal element."
```

Résolution de l'erreur

AWS Organizations les politiques de contrôle des services (SCP) ne prennent pas en charge les `NotPrincipal` éléments `Principal` ou.

Vous pouvez spécifier l'Amazon Resource Name (ARN) à l'aide de la clé de condition globale `aws:PrincipalArn` dans l'élément `Condition`.

Termes connexes

- Syntaxe d'une stratégie de contrôle de service
- Clés de condition globale pour principaux

Erreur : Sid uniques requis

Code d'erreur : UNIQUE_SIDS_REQUIRED

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Unique Sids required: Duplicate statement IDs are not supported for this policy type.
 Update the Sid value.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Duplicate statement IDs are not supported for this policy type.
 Update the Sid value."
```

Résolution de l'erreur

Pour certains types de politique, les ID d'instruction doivent être uniques. L'élément `Sid` (ID d'instruction) vous autorise à saisir un identifiant facultatif que vous pouvez fournir pour l'instruction de politique. Vous pouvez attribuer une valeur d'ID d'instruction à chaque instruction d'un tableau d'instructions à l'aide de l'élément `SID`. Dans un service qui vous permet de spécifier un ID d'élément tel que SQS et SNS, la valeur `Sid` est un simple sous-identifiant de l'ID du document de politique. Par exemple, dans IAM, la valeur `Sid` doit être unique dans une politique JSON.

Termes connexes

- Éléments de politique JSON IAM : Sid

Erreur : action non prise en charge dans la politique

Code d'erreur : UNSUPPORTED_ACTION_IN_POLICY

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Résolution de l'erreur

Certaines actions ne sont pas prises en charge dans l'élément `Action` dans la politique basée sur les ressources, attachée à un type de ressource différent. Par exemple, les AWS Key Management Service actions ne sont pas prises en charge dans les politiques relatives aux compartiments Amazon S3. Spécifiez une action prise en charge par le type de ressource attaché à votre politique basée sur les ressources.

Termes connexes

- Éléments de politique JSON : Action

Erreur : combinaison d'éléments non prise en charge

Code d'erreur : UNSUPPORTED_ELEMENT_COMBINATION

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Unsupported element combination: The policy elements {{element1}} and {{element2}} can
 not be used in the same statement. Remove one of these elements.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The policy elements {{element1}} and {{element2}} can not be used in
 the same statement. Remove one of these elements."
```

Résolution de l'erreur

Certaines combinaisons d'éléments de politique JSON ne peuvent pas être utilisées ensemble. Par exemple, vous ne pouvez pas utiliser `Action` et `NotAction` dans la même instruction de politique. Les autres paires qui s'excluent mutuellement sont notamment `Principal/NotPrincipal` et `Resource/NotResource`.

Termes connexes

- Référence des éléments de politique JSON IAM
- Présentation des politiques JSON

Erreur : clé de condition globale non prise en charge

Code d'erreur : UNSUPPORTED_GLOBAL_CONDITION_KEY

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Unsupported global condition key: The condition key aws:ARN is not supported. Use
 aws:PrincipalArn or aws:SourceArn instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key aws:ARN is not supported. Use aws:PrincipalArn or
 aws:SourceArn instead."
```

Résolution de l'erreur

AWS ne prend pas en charge l'utilisation de la clé de condition globale spécifiée. En fonction de votre cas d'utilisation, vous pouvez utiliser les clés de condition globale `aws:PrincipalArn` ou `aws:SourceArn`. Par exemple, à la place de `aws:ARN`, utilisez `aws:PrincipalArn` pour comparer l'Amazon Resource Name (ARN) du principal ayant fait la demande avec l'ARN spécifié dans la politique. Vous pouvez également utiliser cette clé `aws:SourceArn` globale pour comparer l'Amazon Resource Name (ARN) de la ressource effectuant une demande de service à service avec l'ARN que vous spécifiez dans la politique.

Termes connexes

- AWS clés de contexte de condition globale

Erreur : principal non pris en charge

Code d'erreur : UNSUPPORTED_PRINCIPAL

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Unsupported principal: The policy type {{policy_type}} does not support the Principal
 element. Remove the Principal element.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The policy type {{policy_type}} does not support the Principal
 element. Remove the Principal element."
```

Résolution de l'erreur

L'élément `Principal` spécifie le principal qui est autorisé ou non à accéder à une ressource. Vous ne pouvez pas utiliser l'élément `Principal` dans une politique IAM basée sur l'identité. Vous pouvez l'utiliser dans les politiques de confiance pour les rôles IAM et dans les politiques basées sur les ressources. Resource-based les politiques sont des politiques que vous intégrez directement dans une ressource. Par exemple, vous pouvez intégrer des politiques dans un compartiment Amazon S3 ou une clé AWS KMS.

Termes connexes

- AWS Éléments de politique JSON : principal
- Accès inter-comptes aux ressources dans IAM

Erreur : ARN de ressource non pris en charge dans la politique

Code d'erreur : UNSUPPORTED_RESOURCE_ARN_IN_POLICY

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The resource ARN is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Résolution de l'erreur

Certains ARN de ressources ne sont pas pris en charge dans l'élément `Resource` de la politique basée sur les ressources lorsque cette politique est associée à un autre type de ressource. Par exemple, les AWS KMS ARN ne sont pas pris en charge dans l'`Resource`élément pour les politiques de compartiment Amazon S3. Spécifiez un ARN de ressource pris en charge par un type de ressource attaché à votre politique basée sur les ressources.

Termes connexes

- Éléments de politique JSON : Action

Erreur : sid non pris en charge

Code d'erreur : UNSUPPORTED_SID

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Unsupported Sid: Update the characters in the Sid element to use one of the following
 character types: [a-z, A-Z, 0-9]
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Update the characters in the Sid element to use one of the following
 character types: [a-z, A-Z, 0-9]"
```

Résolution de l'erreur

L'élément `Sid` prend en charge les lettres majuscules, les lettres minuscules et les chiffres.

Termes connexes

- Éléments de politique JSON IAM : Sid

Erreur : caractère générique non pris en charge dans le principal

Code d'erreur : UNSUPPORTED_WILDCARD_IN_PRINCIPAL

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Unsupported wildcard in principal: Wildcards (*, ?) are not supported with the
 principal key {{principal_key}}. Replace the wildcard with a valid principal value.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Wildcards (*, ?) are not supported with the principal key
 {{principal_key}}. Replace the wildcard with a valid principal value."
```

Résolution de l'erreur

La structure de l'élément `Principal` prend en charge l'utilisation d'une paire clé-valeur. La valeur du principal spécifiée dans la politique inclut un caractère générique (*). Vous ne pouvez pas inclure de caractère générique avec la clé de principal que vous avez spécifiée. Par exemple, lorsque vous spécifiez des utilisateurs dans un élément `Principal`, il n'est pas possible d'utiliser un caractère générique signifiant « tous les utilisateurs ». Vous devez nommer un ou plusieurs

utilisateurs spécifiques. De même, lorsque vous spécifiez une session endossed-role, vous ne pouvez pas utiliser un caractère générique pour signifier « toutes les sessions ». Vous devez nommer une session spécifique. Vous ne pouvez pas non plus utiliser de caractère générique pour une correspondance à une partie d'un nom ou d'un ARN.

Pour résoudre ce problème, supprimez le caractère générique et fournissez un principal plus spécifique.

Termes connexes

- AWS Éléments de politique JSON : principal

Erreur : accolade manquante dans la variable

Code d'erreur : MISSING_BRACE_IN_VARIABLE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing brace in variable: The policy variable is missing a closing curly brace. Add }
 after the variable text.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The policy variable is missing a closing curly brace. Add } after
 the variable text."
```

Résolution de l'erreur

La structure de la variable politique prend en charge l'utilisation d'un préfixe $ suivi d'une paire d'accolades ({ }). À l'intérieur des caractères ${ }, incluez le nom de la valeur de la demande que vous souhaitez utiliser dans la politique.

Pour résoudre ce problème, ajoutez l'accolade manquante afin de garantir la présence de l'ensemble d'accolades d'ouverture et de fermeture.

Termes connexes

- Éléments des politiques IAM : Variables

Erreur : guillemet manquant dans la variable

Code d'erreur : MISSING_QUOTE_IN_VARIABLE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing quote in variable: The policy variable default value must begin and end with a
 single quote. Add the missing quote.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The policy variable default value must begin and end with a single
 quote. Add the missing quote."
```

Résolution de l'erreur

Lorsque vous ajoutez une variable à votre politique, vous pouvez spécifier une valeur par défaut pour la variable. Si aucune variable n'est présente, AWS utilise le texte par défaut que vous fournissez.

Pour ajouter une valeur par défaut à une variable, entourez la valeur par défaut de guillemets simples (`' '`), puis séparez le texte de la variable et la valeur par défaut par une virgule et une espace (`, `).

Par exemple, si un principal est labelisé `team=yellow`, il peut accéder au compartiment Amazon S3 `amzn-s3-demo-bucket` avec le nom `amzn-s3-demo-bucket-yellow`. Une politique avec cette ressource peut autoriser les membres de l'équipe à accéder à leurs propres ressources, mais pas à celles d'autres équipes. Pour les utilisateurs sans identifications d'équipe, vous pouvez définir une valeur par défaut de `company-wide`. Ces utilisateurs peuvent accéder uniquement au compartiment `amzn-s3-demo-bucket-company-wide`où ils peuvent afficher des informations générales, telles que des instructions pour rejoindre une équipe.

```
"Resource":"arn:aws:s3:::amzn-s3-demo-bucket-${aws:PrincipalTag/team, 'company-wide'}"
```

Termes connexes

- Éléments des politiques IAM : Variables

Erreur : espace non prise en charge dans la variable

Code d'erreur : UNSUPPORTED_SPACE_IN_VARIABLE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Unsupported space in variable: A space is not supported within the policy variable
 text. Remove the space.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "A space is not supported within the policy variable text. Remove the
 space."
```

Résolution de l'erreur

La structure de la variable politique prend en charge l'utilisation d'un préfixe $ suivi d'une paire d'accolades ({ }). À l'intérieur des caractères ${ }, incluez le nom de la valeur de la demande que vous souhaitez utiliser dans la politique. Bien qu'il soit possible d'inclure une espace lors de la spécification d'une variable par défaut, il n'est pas possible d'en inclure une dans le nom de la variable.

Termes connexes

- Éléments des politiques IAM : Variables

Erreur : variable vide

Code d'erreur : EMPTY_VARIABLE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Empty variable: Empty policy variable. Remove the ${ } variable structure or provide a
 variable within the structure.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Empty policy variable. Remove the ${ } variable structure or provide
 a variable within the structure."
```

Résolution de l'erreur

La structure de la variable politique prend en charge l'utilisation d'un préfixe $ suivi d'une paire d'accolades ({ }). À l'intérieur des caractères ${ }, incluez le nom de la valeur de la demande que vous souhaitez utiliser dans la politique.

Termes connexes

- Éléments des politiques IAM : Variables

Erreur : variable non prise en charge dans l'élément

Code d'erreur : VARIABLE_UNSUPPORTED_IN_ELEMENT

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Variable unsupported in element: Policy variables are supported in the Resource and
 Condition elements. Remove the policy variable {{variable}} from this element.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Policy variables are supported in the Resource and Condition
 elements. Remove the policy variable {{variable}} from this element."
```

Résolution de l'erreur

Vous pouvez utiliser des variables de politique dans l'élément `Resource` et les comparaisons de chaîne de l'élément `Condition`.

Termes connexes

- Éléments des politiques IAM : Variables

Erreur : variable non prise en charge dans la version

Code d'erreur : VARIABLE_UNSUPPORTED_IN_VERSION

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Variable unsupported in version: To include variables in your policy, use the policy
 version 2012-10-17 or later.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "To include variables in your policy, use the policy version
 2012-10-17 or later."
```

Résolution de l'erreur

Pour utiliser des variables de politique, vous devez inclure l'élément `Version` et le définir sur une version qui prend en charge les variables de politique. Les variables ont été introduites dans la version `2012-10-17`. Les versions antérieures du langage de politique ne prennent pas en charge les variables de politique. Si vous ne définissez pas la `Version` sur `2012-10-17` ou ultérieure, des variables telles que `${aws:username}` sont traitées comme des chaînes littérales dans la politique.

Un élément de politique `Version` est différent d'une version de politique. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. Une version de politique est créée lorsque vous apportez des modifications à une politique gérée par le client dans IAM. La politique modifiée ne remplace pas la politique existante. À la place, IAM crée une nouvelle version de la politique gérée.

Termes connexes

- Éléments des politiques IAM : Variables
- Éléments de politique JSON IAM : Version

Erreur : adresse IP privée

Code d'erreur : PRIVATE_IP_ADDRESS

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Private IP address: aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses."
```

Résolution de l'erreur

La clé de condition globale `aws:SourceIp` fonctionne uniquement pour les plages d'adresses IP publiques. Cette erreur s'affiche lorsque votre politique autorise uniquement les adresses IP privées. Dans ce cas, la condition ne correspondrait jamais.

- aws : clé de condition SourceIp globale
- Éléments de politique JSON IAM : Condition

Erreur — Privé NotIpAddress

Code d'erreur : PRIVATE_NOT_IP_ADDRESS

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Private NotIpAddress: The values for condition key aws:SourceIp include only private
 IP addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The values for condition key aws:SourceIp include only private IP
 addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses."
```

Résolution de l'erreur

La clé de condition globale `aws:SourceIp` fonctionne uniquement pour les plages d'adresses IP publiques. Cette erreur s'affiche lorsque vous utilisez l'opérateur de condition `NotIpAddress` et que vous répertoriez uniquement les adresses IP privées. Dans ce cas, la condition correspondrait toujours et serait sans effet.

- aws : clé de condition SourceIp globale
- Éléments de politique JSON IAM : Condition

Erreur : la taille de la politique dépasse le quota SCP

Code d'erreur : POLICY_SIZE_EXCEEDS_SCP_QUOTA

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Policy size exceeds SCP quota: The {{policySize}} characters in the service control
 policy (SCP) exceed the {{policySizeQuota}} character maximum for SCPs. We recommend
 that you use multiple granular policies.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{policySize}} characters in the service control policy (SCP)
 exceed the {{policySizeQuota}} character maximum for SCPs. We recommend that you use
 multiple granular policies."
```

Résolution de l'erreur

AWS Organizations les politiques de contrôle des services (SCP) permettent de spécifier des valeurs dans les `NotAction` éléments `Action` ou. Cependant, ces valeurs peuvent inclure des caractères génériques (*) uniquement à la fin de la chaîne. Cela signifie que vous pouvez spécifier `iam:Get*` mais pas `iam:*role`.

Pour spécifier plusieurs actions, il est AWS recommandé de les répertorier individuellement.

Termes connexes

- Quotas pour les AWS organisations
- AWS Organizations politiques de contrôle des services

Erreur : format de principal de service non valide

Code d'erreur : INVALID_SERVICE_PRINCIPAL_FORMAT

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid service principal format: The service principal does not match the expected
 format. Use the format {{expectedFormat}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The service principal does not match the expected format. Use the
 format {{expectedFormat}}."
```

Résolution de l'erreur

La valeur de la paire clé-valeur de condition doit correspondre à un format de principal de service défini.

Un principal de service est un identifiant utilisé pour accorder des autorisations à un service. Vous pouvez spécifier un principal de service dans l'élément `Principal` ou comme valeur de certaines clés de condition globale et certaines clés spécifiques au service. Le principal de service est défini par chaque service.

L'identifiant d'un principal de service inclut le nom du service, et se présente généralement sous le format suivant, en lettres minuscules :

`service-name``.amazonaws.com`

Certaines clés spécifiques au service peuvent utiliser un format différent pour les principaux de service. Par exemple, la clé de condition `kms:ViaService` requiert le format suivant pour les principaux services, en lettres minuscules :

`service-name.AWS_region``.amazonaws.com`

Termes connexes

- Principaux du service
- AWS clés de condition globales
- `kms:ViaService` clé de condition

Erreur : clé de balise manquante dans la condition

Code d'erreur : MISSING_TAG_KEY_IN_CONDITION

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing tag key in condition: The condition key {{conditionKeyName}} must include a tag
 key to control access based on tags. Use the format {{conditionKeyName}}tag-key and
 specify a key name for tag-key.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key {{conditionKeyName}} must include a tag key to
 control access based on tags. Use the format {{conditionKeyName}}tag-key and specify a
 key name for tag-key."
```

Résolution de l'erreur

Pour contrôler l'accès basé sur des balises, vous devez fournir les informations des balises dans l'élément de condition d'une politique.

Par exemple, pour contrôler l'accès aux AWS ressources, vous incluez la clé de `aws:ResourceTag` condition. Cette clé doit être au format `aws:ResourceTag/`*`tag-key`*. Pour spécifier la clé de balise `owner` et la valeur de balise `JaneDoe` dans une condition, utilisez le format suivant.

```
"Condition": {
    "StringEquals": {"aws:ResourceTag/owner": "JaneDoe"}
}
```

Termes connexes

- Contrôle de l'accès à l'aide de balises
- Conditions
- Clés de condition globale
- AWS clés d'état de service

Erreur : format vpc non valide

Code d'erreur : INVALID_VPC_FORMAT

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid vpc format: The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters."
```

Résolution de l'erreur

La clé de condition `aws:SourceVpc` doit utiliser le préfixe `vpc-` suivi de 8 ou 17 caractères alphanumériques, par exemple, `vpc-11223344556677889` ou `vpc-12345678`.

Termes connexes

- AWS clés de condition globales : aws : SourceVpc

Erreur : format vpce non valide

Code d'erreur : INVALID_VPCE_FORMAT

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid vpce format: The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters."
```

Résolution de l'erreur

La clé de condition `aws:SourceVpce` doit utiliser le préfixe `vpce-` suivi de 8 ou 17 caractères alphanumériques, par exemple, `vpce-11223344556677889` ou `vpce-12345678`.

Termes connexes

- AWS clés de condition globales : aws : SourceVpce

Erreur : principal fédéré non pris en charge

Code d'erreur : FEDERATED_PRINCIPAL_NOT_SUPPORTED

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Federated principal not supported: The policy type does not support a federated
 identity provider in the principal element. Use a supported principal.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The policy type does not support a federated identity provider in
 the principal element. Use a supported principal."
```

Résolution de l'erreur

L'élément `Principal` utilise des principaux fédérés pour les politiques d'approbation attachées aux rôles IAM afin de fournir un accès via la fédération d'identité. Les politiques d'identité et autres politiques basées sur les ressources ne prennent pas en charge un fournisseur d'identité fédéré dans l'élément `Principal`. Par exemple, vous ne pouvez pas utiliser de principal SAML dans une politique de compartiment Amazon S3. Modifiez l'élément `Principal` en un type principal pris en charge.

Termes connexes

- Création d'un rôle pour la fédération d'identité
- Éléments de politique JSON : principal

Erreur : action non prise en charge pour la clé de condition

Code d'erreur : UNSUPPORTED_ACTION_FOR_CONDITION_KEY

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Unsupported action for condition key: The following actions: {{actions}} are not
 supported by the condition key {{key}}. The condition will not be evaluated for these
 actions. We recommend that you move these actions to a different statement without
 this condition key.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The following actions: {{actions}} are not supported by the
 condition key {{key}}. The condition will not be evaluated for these actions. We
 recommend that you move these actions to a different statement without this condition
 key."
```

Résolution de l'erreur

Assurez-vous que la clé de condition dans l'élément `Condition` de l'instruction de politique s'applique à chaque action dans l'élément `Action`. Pour vous assurer que les actions que vous spécifiez sont effectivement autorisées ou rejetées par votre politique, vous devez déplacer les actions non prises en charge vers une autre instruction sans la clé de condition.

> **Note**
>
> Si l'élément `Action` contient des actions avec des caractères génériques, IAM Access Analyzer n'évalue pas ces actions pour cette erreur.

Termes connexes

- Éléments de politique JSON : Action

Erreur : action non prise en charge dans la politique

Code d'erreur : UNSUPPORTED_ACTION_IN_POLICY

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Résolution de l'erreur

Certaines actions ne sont pas prises en charge dans l'élément `Action` dans la politique basée sur les ressources, attachée à un type de ressource différent. Par exemple, les AWS Key Management Service actions ne sont pas prises en charge dans les politiques relatives aux compartiments Amazon S3. Spécifiez une action prise en charge par le type de ressource attaché à votre politique basée sur les ressources.

Termes connexes

- Éléments de politique JSON : Action

Erreur : ARN de ressource non pris en charge dans la politique

Code d'erreur : UNSUPPORTED_RESOURCE_ARN_IN_POLICY

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The resource ARN is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Résolution de l'erreur

Certains ARN de ressources ne sont pas pris en charge dans l'élément `Resource` de la politique basée sur les ressources lorsque cette politique est associée à un autre type de ressource. Par exemple, les AWS KMS ARN ne sont pas pris en charge dans l'`Resource`élément pour les politiques de compartiment Amazon S3. Spécifiez un ARN de ressource pris en charge par un type de ressource attaché à votre politique basée sur les ressources.

Termes connexes

- Éléments de politique JSON : Action

Erreur : clé de condition non prise en charge pour le principal de service

Code d'erreur : UNSUPPORTED_CONDITION_KEY_FOR_SERVICE_PRINCIPAL

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Unsupported condition key for service principal: The following condition keys are not
 supported when used with the service principal: {{conditionKeys}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The following condition keys are not supported when used with the
 service principal: {{conditionKeys}}."
```

Résolution de l'erreur

Vous pouvez spécifier Services AWS dans l'`Principal`élément d'une politique basée sur les ressources à l'aide d'un principal de service, qui est un identifiant pour le service. Vous ne pouvez pas utiliser certaines clés de condition avec certains principaux de service. Par exemple, vous

ne pouvez pas utiliser la clé de condition `aws:PrincipalOrgID` avec le principal de service `cloudfront.amazonaws.com`. Vous devez supprimer les clés de condition qui ne s'appliquent pas au principal de service dans l'élément `Principal`.

Termes connexes

- Principaux du service
- Éléments de politique JSON : principal

Erreur : Erreur de syntaxe de la politique d'approbation notprincipal

Code d'erreur : ROLE_TRUST_POLICY_SYNTAX_ERROR_NOTPRINCIPAL

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Role trust policy syntax error notprincipal: Role trust policies do not support
 NotPrincipal. Update the policy to use a Principal element instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Role trust policies do not support NotPrincipal. Update the policy
 to use a Principal element instead."
```

Résolution de l'erreur

Une politique d'approbation de rôle est une politique basée sur les ressources qui est attachée à un rôle IAM. Les politiques d'approbation définissent quelles entités de principal (comptes, utilisateurs, rôles et utilisateurs fédérés) peuvent endosser le rôle. Les politiques d'approbation de rôle ne prennent pas en charge `NotPrincipal`. Mettez à jour la politique pour utiliser un élément `Principal` à la place.

Termes connexes

- Éléments de politique JSON : principal

- Éléments de politique JSON : NotPrincipal

Erreur : politique d'approbation de rôle générique non pris en charge dans le principal

Code d'erreur : ROLE_TRUST_POLICY_UNSUPPORTED_WILDCARD_IN_PRINCIPAL

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Role trust policy unsupported wildcard in principal: "Principal:" "*" is not supported
 in the principal element of a role trust policy. Replace the wildcard with a valid
 principal value.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": ""Principal:" "*" is not supported in the principal element of a role
 trust policy. Replace the wildcard with a valid principal value."
```

Résolution de l'erreur

Une politique d'approbation de rôle est une politique basée sur les ressources qui est attachée à un rôle IAM. Les politiques d'approbation définissent quelles entités de principal (comptes, utilisateurs, rôles et utilisateurs fédérés) peuvent endosser le rôle. `"Principal:" "*"` n'est pas pris en charge dans l'élément `Principal` d'une politique d'approbation de rôle. Remplacez le caractère générique par une valeur valide du principal.

Termes connexes

- Éléments de politique JSON : principal

Erreur : Erreur de syntaxe de la politique d'approbation resource

Code d'erreur : ROLE_TRUST_POLICY_SYNTAX_ERROR_RESOURCE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Role trust policy syntax error resource: Role trust policies apply to the role
 that they are attached to. You cannot specify a resource. Remove the Resource or
 NotResource element.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Role trust policies apply to the role that they are attached to. You
 cannot specify a resource. Remove the Resource or NotResource element."
```

Résolution de l'erreur

Une politique d'approbation de rôle est une politique basée sur les ressources qui est attachée à un rôle IAM. Les politiques d'approbation définissent quelles entités de principal (comptes, utilisateurs, rôles et utilisateurs fédérés) peuvent endosser le rôle. Les politiques d'approbation de rôles s'appliquent au rôle auquel elles sont associées. Vous ne pouvez pas spécifier un élément `Resource` ou `NotResource` dans une politique d'approbation de rôle. Supprimez l'élément `Resource` ou `NotResource`.

- Éléments de politique JSON : Resource
- Éléments de politique JSON : NotResource

Erreur : non-correspondance de type, plage IP

Code d'erreur : TYPE_MISMATCH_IP_RANGE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Type mismatch IP range: The condition operator {{operator}} is used with an invalid IP
 range value. Specify the IP range in standard CIDR format.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition operator {{operator}} is used with an invalid IP range
 value. Specify the IP range in standard CIDR format."
```

Résolution de l'erreur

Mettez à jour le texte pour utiliser le type de données de l'opérateur de condition Adresse IP, au format CIDR.

Termes connexes

- Opérateurs de condition d'adresse IP
- Éléments de politique JSON IAM : Opérateurs de condition

Erreur : action manquante pour la clé de condition

Code d'erreur : MISSING_ACTION_FOR_CONDITION_KEY

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing action for condition key: The {{actionName}} action must be in the action block
 to allow setting values for the condition key {{keyName}}. Add {{actionName}} to the
 action block.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{actionName}} action must be in the action block to allow
 setting values for the condition key {{keyName}}. Add {{actionName}} to the action
 block."
```

Résolution de l'erreur

La clé de condition dans l'élément `Condition` de la déclaration de politique n'est pas évalué à moins que l'action spécifiée ne figure dans l'élément `Action`. Pour vous assurer que les clés de condition que vous spécifiez sont effectivement autorisées ou rejetées par votre politique, ajoutez l'action à l'élément `Action`.

Termes connexes

- Éléments de politique JSON : Action

Erreur : Syntaxe du principal fédéré non valide dans la politique d'approbation de rôle

Code d'erreur : INVALID_FEDERATED_PRINCIPAL_SYNTAX_IN_ROLE_TRUST_POLICY

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid federated principal syntax in role trust policy: The principal value specifies
 a federated principal that does not match the expected format. Update the federated
 principal to a domain name or a SAML metadata ARN.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The principal value specifies a federated principal that does not
 match the expected format. Update the federated principal to a domain name or a SAML
 metadata ARN."
```

Résolution de l'erreur

La valeur du principal indique un principal fédéré qui ne correspond pas au format attendu. Mettez à jour le format du principal fédéré en utilisant un nom de domaine valide ou un ARN de métadonnées SAML.

Termes connexes

- Utilisateurs fédérés et rôles

Erreur : action non correspondante pour le principal

Code d'erreur : MISMATCHED_ACTION_FOR_PRINCIPAL

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Mismatched action for principal: The {{actionName}} action is invalid with the
 following principal(s): {{principalNames}}. Use a SAML provider principal with
 the sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{actionName}} action is invalid with the following
 principal(s): {{principalNames}}. Use a SAML provider principal with the
 sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options."
```

Résolution de l'erreur

L'action spécifiée dans l'élément `Action` de la déclaration de politique n'est pas valide avec le principal spécifié dans l'élément `Principal`. Par exemple, vous ne pouvez pas utiliser de principal de fournisseur SAML avec l'action `sts:AssumeRoleWithWebIdentity`. Vous devez utiliser un principal de fournisseur SAML avec l'action `sts:AssumeRoleWithSAML` ou un principal de fournisseur OIDC avec l'action `sts:AssumeRoleWithWebIdentity`.

Termes connexes

- AssumeRoleWithSAML
- AssumeRoleWithWebIdentity

Erreur : action manquante pour la politique d'approbation roles anywhere

Code d'erreur : MISSING_ACTION_FOR_ROLES_ANYWHERE_TRUST_POLICY

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing action for roles anywhere trust policy: The rolesanywhere.amazonaws.com service
 principal requires the sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession
 permissions to assume a role. Add the missing permissions to the policy.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The rolesanywhere.amazonaws.com service principal requires the
 sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession permissions to assume a
 role. Add the missing permissions to the policy."
```

Résolution de l'erreur

Pour que Rôles Anywhere IAM soit en mesure d'assumer un rôle et de fournir des informations d'identification AWS temporaires, le rôle doit approuver le principal de service Rôles Anywhere IAM. Le principal de service Rôles Anywhere IAM nécessite les autorisations `sts:AssumeRole`, `sts:SetSourceIdentity`, et `sts:TagSession` pour assumer un rôle. Si l'une des autorisations est manquante, vous devez l'ajouter à votre politique.

Termes connexes

- Modèle de confiance dans Gestion des identités et des accès AWS Roles Anywhere

Erreur : la taille de la politique dépasse le quota RCP

Code d'erreur : POLICY_SIZE_EXCEEDS_RCP_QUOTA

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Policy size exceeds RCP quota: The {{policySize}} characters in the resource control
 policy (RCP) exceed the {{policySizeQuota}} character maximum for RCPs. We recommend
 that you use multiple granular policies.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{policySize}} characters in the resource control policy (RCP)
 exceed the {{policySizeQuota}} character maximum for RCPs. We recommend that you use
 multiple granular policies."
```

Résolution de l'erreur

AWS Organizations les politiques de contrôle des ressources (RCP) permettent de spécifier des valeurs dans l'`Action`élément. Cependant, ces valeurs peuvent inclure des caractères génériques (*) uniquement à la fin de la chaîne. Cela signifie que vous pouvez spécifier `s3:Get*` mais pas `s3:*Object`.

Pour spécifier plusieurs actions, il est AWS recommandé de les répertorier individuellement.

Termes connexes

- Quotas pour AWS Organizations
- AWS Organizations politiques de contrôle des ressources

Erreur : erreur de syntaxe RCP, principal

Code d'erreur : RCP_SYNTAX_ERROR_PRINCIPAL

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
RCP syntax error principal: The Principal element contents are not valid. RCPs only
 support specifying all principals ("*") in the Principal element. The NotPrincipal
 element is not supported for RCPs.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The Principal element contents are not valid. RCPs only support
 specifying all principals ("*") in the Principal element. The NotPrincipal element is
 not supported for RCPs."
```

Résolution de l'erreur

AWS Organizations les politiques de contrôle des ressources (RCP) prennent uniquement en charge la spécification de tous les principaux (» * «) de l'`Principal`élément. L'élément `NotPrincipal` n'est pas pris en charge par les RCP.

Termes connexes

- Syntaxe d'une RCP
- Propriétés du principal

Erreur : erreur de syntaxe RCP, autoriser

Code d'erreur : RCP_SYNTAX_ERROR_ALLOW

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
RCP syntax error allow: RCPs only support specifying all principals ("*") in the
 Principal element, all resources ("*") in the Resource element, and no Condition
 element with an effect of Allow.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "RCPs only support specifying all principals ("*") in the Principal
 element, all resources ("*") in the Resource element, and no Condition element with an
 effect of Allow."
```

Résolution de l'erreur

AWS Organizations les politiques de contrôle des ressources (RCP) prennent uniquement en charge la spécification de tous les principaux (» * «) de l'`Principal`élément et de toutes les ressources (» * «) de l'`Resource`élément. L'élément `Condition` avec un effet `Allow` n'est pas pris en charge par les RCP.

Termes connexes

- Syntaxe d'une RCP
- Propriétés du principal
- Propriétés de la ressource

Erreur : erreur de syntaxe RCP NotAction

Code d'erreur : RCP_SYNTAX_ERROR_NOTACTION

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
RCP syntax error NotAction: RCPs do not support the NotAction element. Update to use
 the Action element.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "RCPs do not support the NotAction element. Update to use the Action
 element."
```

Résolution de l'erreur

AWS Organizations les politiques de contrôle des ressources (RCP) ne prennent pas en charge `NotAction` cet élément. Utilisez l'élément `Action`.

Termes connexes

- Syntaxe d'une RCP
- Éléments de politique JSON IAM : Action
- Éléments de la politique JSON IAM : NotAction

Erreur : erreur de syntaxe RCP, Action

Code d'erreur : RCP_SYNTAX_ERROR_ACTION

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
RCP syntax error action: RCPs only support specifying select service prefixes in the
 Action element. Learn more here.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "RCPs only support specifying select service prefixes in the Action
 element. Learn more here."
```

Résolution de l'erreur

AWS Organizations les politiques de contrôle des ressources (RCP) prennent uniquement en charge la spécification de certains préfixes de service dans l'`Action`élément.

Termes connexes

- Syntaxe d'une RCP
- Liste des Services AWS RCP qui soutiennent

Erreur : compte ARN manquant

Code d'erreur : MISSING_ARN_ACCOUNT

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing ARN account: The resource {{resourceName}} in the arn is missing an account id.
 Please provide a 12 digit account id.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The resource {{resourceName}} in the arn is missing an account id.
 Please provide a 12 digit account id."
```

Résolution de l'erreur

Incluez un ID de compte dans l'ARN de ressource. Les ID de compte sont des nombres entiers à 12 chiffres. Pour savoir comment consulter votre identifiant de compte, voir Trouver votre identifiant de AWS compte.

Termes connexes

- Ressources de politique
- Identificateurs de compte
- ARN de la ressource
- AWS ressources de service avec formats ARN

Erreur : valeur de clé KMS non valide

Code d'erreur : INVALID_KMS_KEY_VALUE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid kms key value: The {{key}} condition key value must be a valid KMS key ARN.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{key}} condition key value must be a valid KMS key ARN."
```

Résolution de l'erreur

Un AWS KMS key ARN (Amazon Resource Name) est un identifiant unique et complet pour une clé KMS. Un ARN clé inclut Compte AWS la région et l'ID de clé. Un ARN de clé suit le format suivant :

`arn:aws:kms:region:account-id:key/key-id`

Termes connexes

- Rechercher l'ID de clé et l'ARN de clé
- ARN clé
- AWS clés de contexte de condition globale

Erreur : utilisation trop permissive de variable

Code d'erreur : VARIABLE_USAGE_TOO_PERMISSIVE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Variable usage too permissive: Overly permissive use of policy variable for the {{key}}
 condition key. Use the policy variable preceded by 6 consecutive characters.
```

```
The policy variable is not allowed in the condition key {{key}}. We consider the key to
 be sensitive and policy variables can be evaluated as effective wildcards. Therefore
 policy variables are not allowed to be used with sensitive keys. Refer to https://
docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_condition-keys.html for the
 list of sensitive keys.
```

```
Overly permissive use of policy variable with aws:userID. Use the policy variable on
 the right side of colon or as the only character on the left side of a colon.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Overly permissive use of policy variable for the {{key}} condition
 key. Use the policy variable preceded by 6 consecutive characters."
```

```
"findingDetails": "The policy variable is not allowed in the condition key {{key}}.
 We consider the key to be sensitive and policy variables can be evaluated as
 effective wildcards. Therefore policy variables are not allowed to be used
```

```
 with sensitive keys. Refer to https://docs.aws.amazon.com/IAM/latest/UserGuide/
reference_policies_condition-keys.html for the list of sensitive keys."
```

```
"findingDetails": "Overly permissive use of policy variable with aws:userID. Use the
 policy variable on the right side of colon or as the only character on the left side
 of a colon."
```

Résolution de l'erreur

Il existe trois messages de résultat possibles pour cette erreur.

- Pour le premier message d'erreur, modifiez l'utilisation de votre variable de politique afin qu'elle soit plus spécifique. Ajoutez au moins six caractères consécutifs avant la variable de politiques afin de réduire la portée des autorisations. Par exemple, au lieu de `${aws:username}`, utilisez `prefix-${aws:username}` ou `myapp-${aws:username}`. Cela garantit que la variable de politique n'accorde pas un accès trop large.
- Pour le deuxième message d'erreur, supprimez la variable de politique de la clé de condition spécifiée. Les variables de politique peuvent agir comme des caractères génériques efficaces et ne sont pas autorisées avec les clés de condition sensibles pour des raisons de sécurité. Utilisez plutôt des valeurs statiques spécifiques ou envisagez de restructurer votre politique afin d'utiliser des clés de condition non sensibles qui prennent en charge les variables de politique.
- Pour le troisième message d'erreur, modifiez l'utilisation de votre variable de politique `aws:userID` afin qu'elle soit plus restrictive. Placez la variable de la politique à droite des deux points (après l'ID du compte) ou utilisez-la comme seul caractère à gauche des deux points. Par exemple, utilisez `AIDACKCEVSQ6C2EXAMPLE:${aws:userid}` ou `${aws:userid}:*` plutôt que `${aws:userid}`.

Termes connexes

- Éléments de politique IAM : variables et balises
- Éléments de politique IAM : condition
- Conditions avec plusieurs clés ou valeurs de contexte
- AWS clés de contexte de condition globale

Erreur : utilisation trop permissive des caractères génériques

Code d'erreur : WILDCARD_USAGE_TOO_PERMISSIVE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Wildcard usage too permissive: Overly permissive use of wildcard for the {{key}}
 condition key. Use the wildcard preceded by 6 consecutive characters.
```

```
The wildcard is not allowed in the condition key {{key}}. We consider the key to
 be sensitive and wildcards are not allowed to be used with sensitive keys. Refer
 to https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_condition-
keys.html for the list of sensitive keys.
```

```
Overly permissive use of wildcard with aws:userID. Use the wildcard on the right side
 of colon or as the only character on the left side of a colon.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Overly permissive use of wildcard for the {{key}} condition key. Use
 the wildcard preceded by 6 consecutive characters."
```

```
"findingDetails": "The wildcard is not allowed in the condition key {{key}}.
 We consider the key to be sensitive and wildcards are not allowed to be used
 with sensitive keys. Refer to https://docs.aws.amazon.com/IAM/latest/UserGuide/
reference_policies_condition-keys.html for the list of sensitive keys."
```

```
"findingDetails": "Overly permissive use of wildcard with aws:userID. Use the wildcard
 on the right side of colon or as the only character on the left side of a colon."
```

Résolution de l'erreur

Il existe trois messages de résultat possibles pour cette erreur.

- Pour le premier message d'erreur, rendez votre utilisation du caractère générique plus spécifique en ajoutant au moins six caractères consécutifs avant le caractère générique. Par exemple, au lieu de `*`, utilisez `prefix-*` ou `prefix-*-suffix`. Cela réduit la portée de la condition et respecte le principe du moindre privilège.

- Pour le deuxième message d'erreur, supprimez le caractère générique de la clé de condition spécifiée. Les caractères génériques sont efficaces et ne sont pas autorisés avec les clés de condition sensibles pour des raisons de sécurité. Remplacez le caractère générique par des valeurs spécifiques correspondant au modèle d'accès souhaité, ou envisagez d'utiliser une clé de condition différente, non sensible, qui prend en charge les caractères génériques.
- Pour le troisième message d'erreur, modifiez l'utilisation de votre caractère générique `aws:userID` afin qu'il soit plus restrictif. Placez le caractère générique à droite des deux points (après l'ID du compte) ou utilisez-le comme seul caractère à gauche des deux points. Par exemple, utilisez `AIDACKCEVSQ6C2EXAMPLE:*` ou `*:*` plutôt que `*`.

Termes connexes

- Éléments de politique IAM : condition
- Conditions avec plusieurs clés ou valeurs de contexte
- AWS clés de contexte de condition globale
- Identifiants IAM

Erreur — DynamoDB ne renvoie pas la capacité consommée

Code de problème : DYNAMODB_INVALID_RETURN_CONSUMED_CAPACITY

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
DynamoDB invalid return consumed capacity: The value for
 dynamodb:ReturnConsumedCapacity is not valid. Update {{value}} to either TOTAL or
 NONE.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The value for dynamodb:ReturnConsumedCapacity is not valid. Update
 {{value}} to either TOTAL or NONE."
```

Résolution de l'erreur

Mettez à jour la valeur de condition en lui attribuant l'une des deux valeurs fournies par DynamoDB : `TOTAL` ou. `NONE` Les valeurs sont sensibles aux majuscules `StringEquals` et minusculesStringLike, donc une variante minuscule telle que `total` n'est pas valide. L'`StringEqualsIgnoreCase`opérateur accepte tous les boîtiers. Une condition qui spécifie une valeur non valide ne correspond à aucune demande. L'instruction n'a donc aucun effet.

Erreur : valeurs renvoyées par DynamoDB non valides

Code de problème : DYNAMODB_INVALID_RETURN_VALUES

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
DynamoDB invalid return values: The value for dynamodb:ReturnValues is not valid.
 Update {{value}} to one of NONE, ALL_OLD, UPDATED_OLD, ALL_NEW, or UPDATED_NEW.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The value for dynamodb:ReturnValues is not valid. Update {{value}}
 to one of NONE, ALL_OLD, UPDATED_OLD, ALL_NEW, or UPDATED_NEW."
```

Résolution de l'erreur

Mettez à jour la valeur de condition en lui attribuant l'une des cinq valeurs fournies par DynamoDB : `NONE`ALL_OLD, `UPDATED_OLD`ALL_NEW, ou. `UPDATED_NEW` Les valeurs sont sensibles aux majuscules `StringEquals` et minusculesStringLike, donc une variante minuscule telle que `updated_new` n'est pas valide. L'`StringEqualsIgnoreCase`opérateur accepte tous les boîtiers. Une condition qui spécifie une valeur non valide ne correspond à aucune demande. L'instruction n'a donc aucun effet.

Erreur — Valeur de sélection non valide de DynamoDB

Code de problème : DYNAMODB_INVALID_SELECT_VALUE

Type de résultat : ERROR

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
DynamoDB invalid select value: The value for dynamodb:Select is not valid. Update
 {{value}} to one of ALL_ATTRIBUTES, ALL_PROJECTED_ATTRIBUTES, SPECIFIC_ATTRIBUTES, or
 COUNT.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The value for dynamodb:Select is not valid. Update {{value}} to one
 of ALL_ATTRIBUTES, ALL_PROJECTED_ATTRIBUTES, SPECIFIC_ATTRIBUTES, or COUNT."
```

Résolution de l'erreur

Mettez à jour la valeur de condition en lui attribuant l'une des quatre valeurs fournies par DynamoDB :`ALL_ATTRIBUTES`, `ALL_PROJECTED_ATTRIBUTESSPECIFIC_ATTRIBUTES`, ou. `COUNT` Les valeurs sont sensibles aux majuscules `StringEquals` et minusculesStringLike, donc une variante minuscule telle que `specific_attributes` n'est pas valide. L'`StringEqualsIgnoreCase`opérateur accepte tous les boîtiers. Une condition qui spécifie une valeur non valide ne correspond à aucune demande. L'instruction n'a donc aucun effet.

Avertissement général — Créez un appareil photo reflex avec NotResource

Code d'erreur : CREATE_SLR_WITH_NOT_RESOURCE

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Create SLR with NotResource: Using the iam:CreateServiceLinkedRole action with
 NotResource can allow creation of unintended service-linked roles for multiple
 resources. We recommend that you specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with NotResource can
 allow creation of unintended service-linked roles for multiple resources. We recommend
 that you specify resource ARNs instead."
```

Résolution de l'avertissement général

L'action `iam:CreateServiceLinkedRole` autorise la création d'un rôle IAM qui permet à un AWS service d'effectuer des actions en votre nom. L'utilisation de `iam:CreateServiceLinkedRole` dans une politique avec l'élément `NotResource` peut autoriser la création de rôles liés au service inattendus pour plusieurs ressources. AWS vous recommande plutôt de spécifier les ARN autorisés dans l'élément `Resource`.

- CreateServiceLinkedRole fonctionnement
- Éléments de la politique JSON IAM : NotResource
- Éléments de politique JSON IAM : Resource

Avertissement général — Créez un reflex avec une étoile en action et NotResource

Code d'erreur : CREATE_SLR_WITH_STAR_IN_ACTION_AND_NOT_RESOURCE

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Create SLR with star in action and NotResource: Using an action with a wildcard(*) and
 NotResource can allow creation of unintended service-linked roles because it can allow
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using an action with a wildcard(*) and NotResource can
 allow creation of unintended service-linked roles because it can allow
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead."
```

Résolution de l'avertissement général

L'action `iam:CreateServiceLinkedRole` autorise la création d'un rôle IAM qui permet à un AWS service d'effectuer des actions en votre nom. Les politiques comportant un caractère générique (*) dans le champ `Action`, et comprenant l'élément `NotResource` peuvent autoriser la création de rôles liés au service inattendus pour plusieurs ressources. AWS vous recommande plutôt de spécifier les ARN autorisés dans l'élément `Resource`.

- CreateServiceLinkedRole fonctionnement
- Éléments de la politique JSON IAM : NotResource
- Éléments de politique JSON IAM : Resource

Avertissement général — Créez un appareil photo reflex avec et NotAction NotResource

Code d'erreur : CREATE_SLR_WITH_NOT_ACTION_AND_NOT_RESOURCE

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Create SLR with NotAction and NotResource: Using NotAction with NotResource
 can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using NotAction with NotResource can allow creation of unintended
 service-linked roles because it allows iam:CreateServiceLinkedRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead."
```

Résolution de l'avertissement général

L'action `iam:CreateServiceLinkedRole` autorise la création d'un rôle IAM qui permet à un AWS service d'effectuer des actions en votre nom. L'utilisation de l'élément `NotAction`

avec l'élément `NotResource` peut autoriser la création de rôles liés au service inattendus pour plusieurs ressources. AWS vous recommande plutôt de réécrire la politique afin d'autoriser `iam:CreateServiceLinkedRole` sur une liste limitée d'ARN dans l'élément `Resource`. Vous pouvez également ajouter `iam:CreateServiceLinkedRole` à l'élément `NotAction`.

- CreateServiceLinkedRole fonctionnement
- Éléments de la politique JSON IAM : NotAction
- Éléments de politique JSON IAM : Action
- Éléments de la politique JSON IAM : NotResource
- Éléments de politique JSON IAM : Resource

Avertissement général : créer un SLR avec star in resource

Code d'erreur : CREATE_SLR_WITH_STAR_IN_RESOURCE

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Create SLR with star in resource: Using the iam:CreateServiceLinkedRole action with
 wildcards (*) in the resource can allow creation of unintended service-linked roles.
 We recommend that you specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with wildcards (*) in
 the resource can allow creation of unintended service-linked roles. We recommend that
 you specify resource ARNs instead."
```

Résolution de l'avertissement général

L'action `iam:CreateServiceLinkedRole` autorise la création d'un rôle IAM qui permet à un AWS service d'effectuer des actions en votre nom. L'utilisation de `iam:CreateServiceLinkedRole` dans une politique avec le caractère générique (*) dans l'élément `Resource` peut autoriser la création de rôles liés au service inattendus pour plusieurs ressources. AWS vous recommande plutôt de spécifier les ARN autorisés dans l'élément `Resource`.

- CreateServiceLinkedRole fonctionnement
- Éléments de politique JSON IAM : Resource

AWS politiques gérées avec cet avertissement général

AWS les politiques gérées vous permettent de commencer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Certains de ces cas d'utilisation sont destinés aux utilisateurs « avec pouvoir » dans votre compte. Les politiques AWS gérées suivantes fournissent un accès aux utilisateurs expérimentés et accordent des autorisations pour créer des rôles liés à un service pour n'importe quel AWS service. AWS vous recommande d'associer les politiques AWS gérées suivantes uniquement aux identités IAM que vous considérez comme des utilisateurs expérimentés.

- PowerUserAccess
- AlexaForBusinessFullAccess
- AWSOrganizationsServiceTrustPolicy— Cette politique AWS gérée fournit des autorisations d'utilisation par le rôle AWS Organizations lié au service. Ce rôle permet aux organisations de créer des rôles liés aux services supplémentaires pour les autres services de votre AWS organisation.

Avertissement général : créer un SLR avec star in action and resource

Code d'erreur : CREATE_SLR_WITH_STAR_IN_ACTION_AND_RESOURCE

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Create SLR with star in action and resource: Using wildcards (*) in the action and
 the resource can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using wildcards (*) in the action and the resource
 can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead."
```

Résolution de l'avertissement général

L'action `iam:CreateServiceLinkedRole` autorise la création d'un rôle IAM qui permet à un AWS service d'effectuer des actions en votre nom. Les politiques comportant un caractère générique (*) dans les éléments `Action` et `Resource` peuvent autoriser la création de rôles liés au service inattendus pour plusieurs ressources. Cela permet de créer un rôle lié à un service lorsque vous spécifiez `"Action": "*""Action": "iam:*"`, ou. `"Action": "iam:Create*"` AWS recommande de spécifier plutôt les ARN autorisés dans l'`Resource`élément.

- CreateServiceLinkedRole fonctionnement
- Éléments de politique JSON IAM : Action
- Éléments de politique JSON IAM : Resource

AWS politiques gérées avec cet avertissement général

AWS les politiques gérées vous permettent de commencer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Certains de ces cas d'utilisation sont destinés aux administrateurs de votre compte. Les politiques AWS gérées suivantes fournissent un accès administrateur et accordent des autorisations pour créer des rôles liés à un service pour n'importe quel AWS service. AWS vous recommande d'associer les politiques AWS gérées suivantes uniquement aux identités IAM que vous considérez comme des administrateurs.

- AdministratorAccess
- IAMFullAccess

Avertissement général — Créez un reflex avec une étoile dans la ressource et NotAction

Code d'erreur : CREATE_SLR_WITH_STAR_IN_RESOURCE_AND_NOT_ACTION

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Create SLR with star in resource and NotAction: Using a resource with wildcards (*)
 and NotAction can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can allow creation
 of unintended service-linked roles because it allows iam:CreateServiceLinkedRole
 permissions on all resources. We recommend that you specify resource ARNs instead."
```

Résolution de l'avertissement général

L'action `iam:CreateServiceLinkedRole` autorise la création d'un rôle IAM qui permet à un AWS service d'effectuer des actions en votre nom. L'utilisation de l'élément `NotAction` dans une politique avec le caractère générique (*) dans l'élément `Resource` peut autoriser la création de rôles liés au service inattendus pour plusieurs ressources. AWS vous recommande plutôt de spécifier les ARN autorisés dans l'élément `Resource`. Vous pouvez également ajouter `iam:CreateServiceLinkedRole` à l'élément `NotAction`.

- CreateServiceLinkedRole fonctionnement
- Éléments de la politique JSON IAM : NotAction
- Éléments de politique JSON IAM : Action
- Éléments de politique JSON IAM : Resource

Avertissement général : clé de condition globale obsolète

Code d'erreur : DEPRECATED_GLOBAL_CONDITION_KEY

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Deprecated global condition key: We recommend that you update aws:ARN to use the newer
 condition key aws:PrincipalArn.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "We recommend that you update aws:ARN to use the newer condition key
 aws:PrincipalArn."
```

Résolution de l'avertissement général

La politique inclut une clé de condition globale obsolète. Mettez à jour la clé de condition dans la paire clé-valeur de condition pour utiliser une clé de condition globale prise en charge.

- Clés de condition globale

Avertissement général : valeur de date non valide

Code d'erreur : INVALID_DATE_VALUE

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid date value: The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format."
```

Résolution de l'avertissement général

L'heure d'une époque Unix décrit un point dans le temps qui s'est écoulé depuis le 1er janvier 1970, moins les secondes intercalaires. L'heure d'Epoch peut ne pas correspondre à l'heure précise

à laquelle vous vous attendez. AWS recommande d'utiliser la norme W3C pour les formats de date et d'heure. Par exemple, vous pouvez spécifier une date complète, telle que YYYY-MM-DD (1997-07-16), ou vous pouvez également ajouter l'heure à la seconde, telle que YYYY-MM-DDThh:mm:ssTZD (T19:20:301997-07-16 + 01:00).

- Formats de date et d'heure selon la norme W3C
- Éléments de politique JSON IAM : Version
- aws : clé de condition CurrentTime globale

Avertissement général : référence de rôle non valide

Code d'erreur : INVALID_ROLE_REFERENCE

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid role reference: The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead."
```

Résolution de l'avertissement général

AWS recommande de spécifier l'Amazon Resource Name (ARN) pour un rôle IAM au lieu de son ID principal. Lorsque IAM enregistre la politique, il transforme l'ARN en identifiant principal pour le rôle existant. AWS comprend une mesure de sécurité. Si quelqu'un supprime le rôle et le recrée, il aura un nouvel ID, et la politique ne correspondra pas à l'ID du nouveau rôle.

- Spécification d'un principal : rôles IAM
- ARN IAM
- ID uniques IAM

Avertissement général : référence utilisateur non valide

Code d'erreur : INVALID_USER_REFERENCE

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Invalid user reference: The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead."
```

Résolution de l'avertissement général

AWS recommande de spécifier l'Amazon Resource Name (ARN) pour un utilisateur IAM au lieu de son ID principal. Lorsque IAM enregistre la politique, il transforme l'ARN en identifiant principal de l'utilisateur existant. AWS comprend une mesure de sécurité. Si quelqu'un supprime l'utilisateur et le recrée, il aura un nouvel ID, et la politique ne correspondra pas à l'ID du nouvel utilisateur.

- Spécification d'un principal : utilisateurs IAM
- ARN IAM
- ID uniques IAM

Avertissement général : version manquante

Code d'erreur : MISSING_VERSION

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing version: We recommend that you specify the Version element to help you with
 debugging permission issues.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "We recommend that you specify the Version element to help you with
 debugging permission issues."
```

Résolution de l'avertissement général

AWS vous recommande d'inclure le `Version` paramètre facultatif dans votre politique. Si vous n'incluez pas d'élément Version, la valeur par défaut est `2012-10-17`, mais de nouvelles fonctions, telles que des variables de politique, ne fonctionneront pas avec votre politique. Par exemple, les variables telles que `${aws:username}` ne sont pas reconnues et sont traitées comme des chaînes littérales dans la politique.

- Éléments de politique JSON IAM : Version

Avertissement général : sid uniques recommandés

Code d'erreur : UNIQUE_SIDS_RECOMMENDED

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Unique Sids recommended: We recommend that you use statement IDs that are unique to
 your policy. Update the Sid value.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "We recommend that you use statement IDs that are unique to your
 policy. Update the Sid value."
```

Résolution de l'avertissement général

AWS vous recommande d'utiliser des identifiants de déclaration uniques. L'élément `Sid` (ID d'instruction) vous autorise à saisir un identifiant facultatif que vous pouvez fournir pour l'instruction de politique. Vous pouvez attribuer une valeur d'ID d'instruction à chaque instruction d'un tableau d'instructions à l'aide de l'élément `SID`.

Termes connexes

- Éléments de politique JSON IAM : Sid

Avertissement général : caractère générique sans opérateur similaire

Code d'erreur : WILDCARD_WITHOUT_LIKE_OPERATOR

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Wildcard without like operator: Your condition value includes a * or ? character. If
 you meant to use a wildcard (*, ?), update the condition operator to include Like.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Your condition value includes a * or ? character. If you meant to
 use a wildcard (*, ?), update the condition operator to include Like."
```

Résolution de l'avertissement général

La structure de l'élément `Condition` a besoin que vous utilisiez un opérateur de condition et une paire clé-valeur. Lorsque vous spécifiez une valeur de condition qui utilise un caractère générique (*, ?), vous devez utiliser la version `Like` de l'opérateur de condition. Par exemple, au lieu de l'opérateur de condition de chaîne `StringEquals`, utilisez `StringLike`.

```
"Condition": {"StringLike": {"aws:PrincipalTag/job-category": "admin-*"}}
```

- Éléments de politique JSON IAM : Opérateurs de condition
- Éléments de politique JSON IAM : Condition

AWS politiques gérées avec cet avertissement général

AWS les politiques gérées vous permettent de commencer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Les politiques AWS gérées suivantes incluent des caractères génériques dans leur valeur de condition sans opérateur de condition inclus `Like` pour la mise en correspondance des modèles. Lorsque vous utilisez la politique AWS gérée comme référence pour créer votre politique gérée par le client, vous AWS recommande d'utiliser un opérateur de condition qui prend en charge la correspondance de modèles avec des caractères génériques (*, ?) , tels que`StringLike`.

- AWSGlueConsoleSageMakerNotebookFullAccess

Avertissement général : la taille de la politique dépasse le quota de politique

Code d'erreur : POLICY_SIZE_EXCEEDS_IDENTITY_POLICY_QUOTA

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Policy size exceeds identity policy quota: The {{policySize}} characters in the
 identity policy, excluding whitespace, exceed the {{policySizeQuota}} character
 maximum for inline and managed policies. We recommend that you use multiple granular
 policies.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{policySize}} characters in the identity policy, excluding
 whitespace, exceed the {{policySizeQuota}} character maximum for inline and managed
 policies. We recommend that you use multiple granular policies."
```

Résolution de l'avertissement général

Vous pouvez associer jusqu'à 20 politiques gérées à une identité IAM (utilisateur, groupe d'utilisateurs ou rôle). La taille de chaque politique gérée ne peut toutefois pas dépasser le quota par défaut de 6 144 caractères. IAM ne compte pas les espaces lors du calcul de la taille d'une politique

par rapport à ce quota. Les quotas, également appelés limites dans AWS, sont les valeurs maximales pour les ressources, les actions et les objets de votre AWS compte.

Vous pouvez également ajouter autant de politiques en ligne que vous le voulez à une identité IAM. Cependant, la somme de toutes les politiques en ligne par identité ne peut pas dépasser le quota spécifié.

Si votre politique dépasse le quota, vous pouvez organiser votre politique en plusieurs instructions et les regrouper en plusieurs politiques.

Termes connexes

- IAM et quotas de AWS STS caractères
- Plusieurs instructions et politiques
- Politiques gérées par le client IAM
- Présentation des politiques JSON
- Syntaxe de politique IAM JSON

AWS politiques gérées avec cet avertissement général

AWS les politiques gérées vous permettent de commencer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Les politiques AWS gérées suivantes accordent des autorisations à des actions sur de nombreux AWS services et dépassent la taille maximale de la stratégie. Lorsque vous utilisez la politique gérée par AWS comme référence pour créer votre politique gérée, vous devez diviser la politique en plusieurs politiques.

- ReadOnlyAccess
- AWSSupportServiceRolePolicy

Avertissement général : la taille de la politique dépasse le quota de la politique de ressources

Code d'erreur : POLICY_SIZE_EXCEEDS_RESOURCE_POLICY_QUOTA

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Policy size exceeds resource policy quota: The {{policySize}} characters in the
 resource policy exceed the {{policySizeQuota}} character maximum for resource
 policies. We recommend that you use multiple granular policies.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{policySize}} characters in the resource policy exceed the
 {{policySizeQuota}} character maximum for resource policies. We recommend that you use
 multiple granular policies."
```

Résolution de l'avertissement général

Resource-based les politiques sont des documents de stratégie JSON que vous joignez à une ressource, telle qu'un compartiment Amazon S3. Ces politiques accordent au principal spécifié l'autorisation d'effectuer des actions spécifiques sur cette ressource et définissent sous quelles conditions cela s'applique. La taille des politiques basées sur les ressources ne peut pas dépasser le quota défini pour cette ressource. Les quotas, également appelés limites dans AWS, sont les valeurs maximales pour les ressources, les actions et les objets de votre AWS compte.

Si votre politique dépasse le quota, vous pouvez organiser votre politique en plusieurs instructions et les regrouper en plusieurs politiques.

Termes connexes

- Resource-based politiques
- Politiques de compartiment Amazon S3
- Plusieurs instructions et politiques
- Présentation des politiques JSON
- Syntaxe de politique IAM JSON

Avertissement général : non-correspondance de type

Code d'erreur : TYPE_MISMATCH

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Type mismatch: Use the operator type {{allowed}} instead of operator {{operator}} for
 the condition key {{key}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Use the operator type {{allowed}} instead of operator {{operator}}
 for the condition key {{key}}."
```

Résolution de l'avertissement général

Mettez à jour le texte pour utiliser le type de données de l'opérateur de condition pris en charge.

Par exemple, la clé de condition globale `aws:MultiFactorAuthPresent` a besoin d'un opérateur de condition avec le type de données `Boolean`. Si vous fournissez une date ou un entier, le type de données ne correspondra pas.

Termes connexes

- Clés de condition globale
- Éléments de politique JSON IAM : Opérateurs de condition

Avertissement général : non-correspondance de type, opérateur booléen

Code d'erreur : TYPE_MISMATCH_BOOLEAN

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Type mismatch Boolean: Add a valid Boolean value (true or false) for the condition
 operator {{operator}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a valid Boolean value (true or false) for the condition operator
 {{operator}}."
```

Résolution de l'avertissement général

Mettez à jour le texte pour utiliser un type de données d'opérateur de condition booléen, tel que `true` ou `false`.

Par exemple, la clé de condition globale `aws:MultiFactorAuthPresent` a besoin d'un opérateur de condition avec le type de données `Boolean`. Si vous fournissez une date ou un entier, le type de données ne correspondra pas.

Termes connexes

- Opérateurs de condition booléens
- Éléments de politique JSON IAM : Opérateurs de condition

Avertissement général : non-correspondance de type, date

Code d'erreur : TYPE_MISMATCH_DATE

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Type mismatch date: The date condition operator is used with an invalid value. Specify
 a valid date using YYYY-MM-DD or other ISO 8601 date/time format.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The date condition operator is used with an invalid value. Specify a
 valid date using YYYY-MM-DD or other ISO 8601 date/time format."
```

Résolution de l'avertissement général

Mettez à jour le texte pour utiliser le type de données de l'opérateur de condition Date, dans un format d'heure/date YYYY-MM-DD ou un autre format d'heure/date ISO 8601.

Termes connexes

- Opérateurs de condition de date
- Éléments de politique JSON IAM : Opérateurs de condition

Avertissement général : numéro de non-correspondance de type

Code d'erreur : TYPE_MISMATCH_NUMBER

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Type mismatch number: Add a valid numeric value for the condition operator
 {{operator}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a valid numeric value for the condition operator {{operator}}."
```

Résolution de l'avertissement général

Mettez à jour le texte pour utiliser le type de données de l'opérateur de condition numérique.

Termes connexes

- Opérateurs de condition numériques
- Éléments de politique JSON IAM : Opérateurs de condition

Avertissement général : non-correspondance de type, chaîne

Code d'erreur : TYPE_MISMATCH_STRING

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Type mismatch string: Add a valid base64-encoded string value for the condition
 operator {{operator}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a valid base64-encoded string value for the condition operator
 {{operator}}."
```

Résolution de l'avertissement général

Mettez à jour le texte pour utiliser le type de données de l'opérateur de condition Chaîne.

Termes connexes

- Opérateurs de condition de chaîne
- Éléments de politique JSON IAM : Opérateurs de condition

Avertissement général : github repo et branch spécifiques recommandés

Code d'erreur : SPECIFIC_GITHUB_REPO_AND_BRANCH_RECOMMENDED

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Specific github repo and branch recommended: Using a wildcard (*) in
 token.actions.githubusercontent.com:sub can allow requests from more sources than
 you intended. Specify the value of token.actions.githubusercontent.com:sub with the
 repository and branch name. If the subject claim does not include the branch, specify
 it in a token.actions.githubusercontent.com:ref condition key.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using a wildcard (*) in token.actions.githubusercontent.com:sub
 can allow requests from more sources than you intended. Specify the value
 of token.actions.githubusercontent.com:sub with the repository and branch
 name. If the subject claim does not include the branch, specify it in a
 token.actions.githubusercontent.com:ref condition key."
```

Résolution de l'avertissement général

Si vous l'utilisez GitHub en tant qu'IdP OIDC, la meilleure pratique consiste à limiter le nombre d'entités pouvant assumer le rôle associé à l'IdP IAM. Lorsque vous incluez une `Condition` déclaration dans une politique de confiance relative aux rôles, vous pouvez limiter le rôle à une GitHub organisation, un référentiel ou une branche spécifique. Vous pouvez utiliser la clé de condition `token.actions.githubusercontent.com:sub` pour limiter l'accès. Nous vous recommandons de limiter la condition à un ensemble spécifique de référentiels ou de branches. Si vous utilisez un caractère générique (`*`) dans`token.actions.githubusercontent.com:sub`, les GitHub actions émanant d'organisations ou de référentiels indépendants de votre volonté peuvent assumer les rôles associés à l'IdP GitHub IAM de votre compte. AWS

Spécifiez le référentiel dans`token.actions.githubusercontent.com:sub`.
Vous pouvez spécifier la branche avec cette même valeur ou dans une clé de `token.actions.githubusercontent.com:ref` condition distincte. Lorsque l'objet de la revendication n'inclut pas la branche, par exemple lorsqu'elle nomme un GitHub environnement, spécifiez la branche dans`token.actions.githubusercontent.com:ref`.

Termes connexes

- Configuration d'un rôle pour le fournisseur d'identité GitHub OIDC

Avertissement général : la taille de la politique dépasse le quota de la politique de rôle

Code d'erreur : POLICY_SIZE_EXCEEDS_ROLE_TRUST_POLICY_QUOTA

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Policy size exceeds role trust policy quota: The characters in the role trust policy,
 excluding whitespace, exceed the character maximum. We recommend that you request a
```

```
role trust policy length quota increase using Service Quotas and AWS Support Center.
If the quotas have already been increased, then you can ignore this warning.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The characters in the role trust policy, excluding whitespace,
 exceed the character maximum. We recommend that you request a role trust policy length
 quota increase using Service Quotas and AWS Support Center. If the quotas have already
 been increased, then you can ignore this warning."
```

Résolution de l'avertissement général

IAM et définissez AWS STS des quotas qui limitent la taille des politiques de confiance relatives aux rôles. Les caractères de la politique d'approbation de rôle, à l'exception des espaces, dépassent le nombre maximum de caractères. Nous vous recommandons de demander une augmentation du quota de durée de la politique d'approbation de rôle en utilisant Service Quotas et AWS Support Center Console.

Termes connexes

- IAM et AWS STS quotas, exigences relatives aux noms et limites de caractères

Avertissement général : la clé de condition principale associée au RCP est manquante

Code d'erreur : RCP_MISSING_RELATED_PRINCIPAL_CONDITION_KEY

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
RCP missing related principal condition key: RCPs impact IAM roles, users, and AWS
 service principals. To prevent unintended impact to services acting on your behalf
 using a service principal, an additional statement should be added to the Condition
 block "BoolIfExists": { "aws:PrincipalIsAWSService": "false"} whenever a principal key
 {{conditionKeyName}} is used.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "RCPs impact IAM roles, users, and AWS service principals. To prevent
 unintended impact to services acting on your behalf using a service principal,
 an additional statement should be added to the Condition block "BoolIfExists":
 { "aws:PrincipalIsAWSService": "false"} whenever a principal key {{conditionKeyName}}
 is used."
```

Résolution de l'avertissement général

AWS Organizations les politiques de contrôle des ressources (RCP) peuvent avoir un impact sur les rôles, les utilisateurs et Service AWS les directeurs IAM. Pour éviter tout impact imprévu sur les services agissant en votre nom à l'aide d'un principal de service, ajoutez la déclaration suivante à votre élément `Condition` :

```
"BoolIfExists": { "aws:PrincipalIsAWSService": "false"}
```

Termes connexes

- Syntaxe d'une RCP
- Propriétés du principal

Avertissement général : la clé de condition du principal de service associée au RCP est manquante

Code d'erreur : RCP_MISSING_RELATED_SERVICE_PRINCIPAL_CONDITION_KEY

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
RCP missing related service principal condition key: RCPs impact IAM roles, users,
 and AWS service principals. To prevent unintended impact to your principals,
 an additional statement should be added to the Condition block "BoolIfExists":
 { "aws:PrincipalIsAWSService": "true"} whenever the key {{conditionKeyName}} is used.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "RCPs impact IAM roles, users, and AWS service principals. To prevent
 unintended impact to your principals, an additional statement should be added to the
 Condition block "BoolIfExists": { "aws:PrincipalIsAWSService": "true"} whenever the
 key {{conditionKeyName}} is used."
```

Résolution de l'avertissement général

AWS Organizations les politiques de contrôle des ressources (RCP) peuvent avoir un impact sur les rôles, les utilisateurs et Service AWS les directeurs IAM. Pour éviter tout impact imprévu sur vos principaux, ajoutez l'instruction suivante à votre élément `Condition` :

```
"BoolIfExists": { "aws:PrincipalIsAWSService": "true"}
```

Termes connexes

- Syntaxe d'une RCP
- Propriétés du principal

Avertissement général : vérification nulle de la clé de condition de service manquante pour le RCP

Code d'erreur : RCP_MISSING_SERVICE_CONDITION_KEY_NULL_CHECK

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
RCP missing service condition key null check: The specified service may have a service
 integration that does not require the use of the {{conditionKeyName}} condition
 key. To prevent unintended impact to services acting on your behalf using a service
 principal, an additional statement should be added to the Condition block "Null":
 { "aws:SourceAccount": "false"} or "Null": { "aws:SourceArn": "false"} whenever the
 key {{conditionKeyName}} is used.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The specified service may have a service integration that does
 not require the use of the {{conditionKeyName}} condition key. To prevent unintended
 impact to services acting on your behalf using a service principal, an additional
 statement should be added to the Condition block "Null": { "aws:SourceAccount":
 "false"} or "Null": { "aws:SourceArn": "false"} whenever the key {{conditionKeyName}}
 is used."
```

Résolution de l'avertissement général

AWS Organizations les politiques de contrôle des ressources (RCP) peuvent avoir un impact sur les rôles, les utilisateurs et Service AWS les directeurs IAM. Pour éviter tout impact imprévu sur les services agissant en votre nom à l'aide d'un principal de service, ajoutez l'une des instructions suivantes à votre élément `Condition` chaque fois que la clé spécifiée est utilisée :

```
"Null": { "aws:SourceAccount": "false"}
```

or

```
"Null": { "aws:SourceArn": "false"}
```

Termes connexes

- Syntaxe d'une RCP
- Propriétés du principal

Avertissement général : utilisez la clé de condition uniquement avec les services pris en charge.

Code d'erreur : USE_CONDITION_KEY_ONLY_WITH_SUPPORTED_SERVICES

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Use condition key only with supported services: The condition key {{key}} works only
 with specific AWS services and must be scoped to supported services in your policies.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key {{key}} works only with specific AWS services and
 must be scoped to supported services in your policies.
```

Résolution de l'avertissement général

Consultez la AWS documentation pour identifier celles qui Services AWS soutiennent cette clé de condition. Si l'un des services de votre politique ne prend pas en charge la clé de condition, modifiez votre politique pour limiter la clé de condition à ceux Services AWS qui la prennent en charge.

Termes connexes

- `aws:VpceAccount`
- `aws:VpceOrgID`
- `aws:VpceOrgPaths`
- Actions, ressources et clés de condition pour Services AWS

Avertissement général — DynamoDB remplace l'opération par des attributs

Code de problème : DYNAMODB_REPLACE_OPERATION_WITH_ATTRIBUTES

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
DynamoDB replace operation with attributes: The {{action}} action replaces the entire
 item, so dynamodb:Attributes cannot restrict access at the attribute level. That
 condition only applies to UpdateItem; use dynamodb:UpdateItem for attribute-level
 access control.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{action}} action replaces the entire item, so
 dynamodb:Attributes cannot restrict access at the attribute level. That condition only
 applies to UpdateItem; use dynamodb:UpdateItem for attribute-level access control."
```

Résolution de l'avertissement général

La `dynamodb:Attributes` condition ne limite pas `dynamodb:PutItem` ou`dynamodb:BatchWriteItem`. Ces actions remplacent l'ensemble de l'élément, de sorte qu'un appelant peut supprimer un attribut restreint simplement en l'omettant de l'élément qu'il écrit. Pour contrôler les attributs qu'un appelant peut modifier, accordez à la `dynamodb:UpdateItem` place et conservez la `dynamodb:Attributes` condition sur cette instruction. Si vous n'aviez pas l'intention de contrôler le niveau des attributs ici, supprimez la `dynamodb:Attributes` condition.

Avertissement général : le scan DynamoDB contourne les touches principales

Code de problème : DYNAMODB_SCAN_BYPASSES_LEADING_KEYS

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
DynamoDB scan bypasses leading keys: The dynamodb:Scan operation returns all items
 in the table, bypassing the dynamodb:LeadingKeys restriction. Consider removing
 dynamodb:Scan.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The dynamodb:Scan operation returns all items in the table,
 bypassing the dynamodb:LeadingKeys restriction. Consider removing dynamodb:Scan."
```

Résolution de l'avertissement général

`dynamodb:Scan`lit la totalité du tableau, de sorte que la `dynamodb:LeadingKeys` condition de cette instruction ne limite pas les éléments qu'un appelant peut lire. `dynamodb:Scan`Supprimez-le ou accordez-le dans une déclaration séparée qui ne repose pas sur`dynamodb:LeadingKeys`.

Avertissement général — DynamoDB n'est pas la bonne condition pour agir

Code de problème : DYNAMODB_WRONG_CONDITION_FOR_ACTION

Type de résultat : GENERAL_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
DynamoDB wrong condition for action: The condition key {{conditionKey}} does
 not apply to the {{action}} action. Use dynamodb:Select with read actions and
 dynamodb:ReturnValues with write actions.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key {{conditionKey}} does not apply to the {{action}}
 action. Use dynamodb:Select with read actions and dynamodb:ReturnValues with write
 actions."
```

Résolution de l'avertissement général

Cette instruction est utilisée `dynamodb:Select` avec une action d'écriture ou `dynamodb:ReturnValues` avec une action de lecture. DynamoDB fournit `dynamodb:Select` uniquement des actions de lecture et `dynamodb:ReturnValues` uniquement des actions d'écriture, de sorte que la condition ici ne prend jamais effet et ne protège rien. Associez chaque touche à une action complémentaire ou retirez-la. Cette vérification s'applique à la fois aux `Deny` déclarations `Allow` et.

Avertissement de sécurité : clé de condition non fiable

Code d’erreur : UNTRUSTWORTHY_CONDITION_KEY

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Untrustworthy condition key: The {{key}} condition key is not recommended for access
 control as it can be spoofed/manipulated by the caller.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{key}} condition key is not recommended for access control as
 it can be spoofed/manipulated by the caller."
```

Résolution de l'avertissement de sécurité

N'utilisez pas cette clé de condition pour le contrôle d'accès. L'appelant peut potentiellement manipuler ou usurper la valeur de la clé, ce qui crée un risque pour la sécurité.

Termes connexes

- AWS clés de contexte de condition globale

Avertissement de sécurité — Autoriser avec NotPrincipal

Code d'erreur : ALLOW_WITH_NOT_PRINCIPAL

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Allow with NotPrincipal: Using Allow with NotPrincipal can be overly permissive. We
 recommend that you use Principal instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using Allow with NotPrincipal can be overly permissive. We recommend
 that you use Principal instead."
```

Résolution de l'avertissement de sécurité

L'utilisation de `"Effect": "Allow"` avec `NotPrincipal` peut être trop permissive. Par exemple, cela peut accorder des autorisations à des mandants anonymes. AWS vous recommande de spécifier les principaux qui ont besoin d'accéder à l'aide de l'`Principal`élément. En variante, vous pouvez autoriser un accès étendu, puis ajouter une autre instruction qui utilise l'élément `NotPrincipal` avec `“Effect”: “Deny”`.

- AWS Éléments de politique JSON : principal
- AWS Éléments de politique JSON : NotPrincipal

Avertissement de sécurité — ForAllValues avec clé à valeur unique

Code d'erreur : FORALLVALUES_WITH_SINGLE_VALUED_KEY

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
ForAllValues with single valued key: Using ForAllValues qualifier with the single-
valued condition key {{key}} can be overly permissive. We recommend that you remove
 ForAllValues:.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using ForAllValues qualifier with the single-valued condition key
 {{key}} can be overly permissive. We recommend that you remove ForAllValues:."
```

Résolution de l'avertissement de sécurité

AWS recommande de n'utiliser `ForAllValues` que les conditions comportant plusieurs valeurs. L'opération d'ensemble `ForAllValues` teste si la valeur de chaque membre de la requête est un sous-ensemble de la clé de condition. La condition renvoie la valeur true si chaque valeur de clé de la demande correspond à au moins une valeur de la politique. Elle renvoie également la valeur Vrai si la demande ne comprend pas de clés ou si les valeurs de clé aboutissent à un ensemble de données nul, tel qu'une chaîne vide.

Pour savoir si une condition prend en charge une valeur unique ou plusieurs valeurs, passez en revue la page Actions, ressources et clés de condition pour le service. Les clés de condition avec le préfixe de type de données `ArrayOf` sont des clés de condition à valeurs multiples. Par exemple, Amazon SES prend en charge les clés avec des valeurs uniques (`String`) et le type de données à valeurs multiples `ArrayOfString`.

- Clés de contexte à valeurs multiples

Avertissement de sécurité — Passez le rôle avec NotResource

Code d'erreur : PASS_ROLE_WITH_NOT_RESOURCE

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Pass role with NotResource: Using the iam:PassRole action with NotResource can be
 overly permissive because it can allow iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the iam:PassRole action with NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

Résolution de l'avertissement de sécurité

Pour configurer de nombreux AWS services, vous devez leur transmettre un rôle IAM. Pour permettre cela, vous devez accorder l'autorisation `iam:PassRole` à une identité (utilisateur, groupe d'utilisateurs ou rôle). L'utilisation de l'`NotResource`élément `iam:PassRole` dans une politique peut permettre à vos clients d'accéder à plus de services ou de fonctionnalités que vous ne le souhaitiez. AWS recommande de spécifier plutôt les ARN autorisés dans l'`Resource`élément. En outre, vous pouvez réduire les autorisations à un seul service en utilisant la clé de condition `iam:PassedToService`.

- Transférer un rôle à un service
- iam : PassedToService
- Éléments de la politique JSON IAM : NotResource
- Éléments de politique JSON IAM : Resource

Avertissement de sécurité — Rôle de passe avec une étoile en action et NotResource

Code d'erreur : PASS_ROLE_WITH_STAR_IN_ACTION_AND_NOT_RESOURCE

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Pass role with star in action and NotResource: Using an action with a wildcard (*) and
 NotResource can be overly permissive because it can allow iam:PassRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using an action with a wildcard (*) and NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

Résolution de l'avertissement de sécurité

Pour configurer de nombreux AWS services, vous devez leur transmettre un rôle IAM. Pour permettre cela, vous devez accorder l'autorisation `iam:PassRole` à une identité (utilisateur, groupe d'utilisateurs ou rôle). Les politiques comportant un caractère générique (*) `Action` et qui incluent l'`NotResource`élément peuvent permettre à vos mandants d'accéder à plus de services ou de fonctionnalités que vous ne le souhaitiez. AWS recommande de spécifier plutôt les ARN autorisés dans l'`Resource`élément. En outre, vous pouvez réduire les autorisations à un seul service en utilisant la clé de condition `iam:PassedToService`.

- Transférer un rôle à un service
- iam : PassedToService
- Éléments de la politique JSON IAM : NotResource
- Éléments de politique JSON IAM : Resource

Avertissement de sécurité — Rôle de passe avec NotAction et NotResource

Code d'erreur : PASS_ROLE_WITH_NOT_ACTION_AND_NOT_RESOURCE

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Pass role with NotAction and NotResource: Using NotAction with NotResource can
 be overly permissive because it can allow iam:PassRole permissions on multiple
 resources.. We recommend that you specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using NotAction with NotResource can be overly permissive because
 it can allow iam:PassRole permissions on multiple resources.. We recommend that you
 specify resource ARNs instead."
```

Résolution de l'avertissement de sécurité

Pour configurer de nombreux AWS services, vous devez leur transmettre un rôle IAM. Pour permettre cela, vous devez accorder l'autorisation `iam:PassRole` à une identité (utilisateur, groupe d'utilisateurs ou rôle). L'utilisation de l'`NotAction`élément et la liste de certaines ressources qu'il contient peuvent permettre à vos directeurs d'accéder à plus de services ou de fonctionnalités que vous ne le souhaitiez. `NotResource` AWS recommande de spécifier plutôt les ARN autorisés dans l'`Resource`élément. En outre, vous pouvez réduire les autorisations à un seul service en utilisant la clé de condition `iam:PassedToService`.

- Transférer un rôle à un service
- iam : PassedToService
- Éléments de la politique JSON IAM : NotAction
- Éléments de politique JSON IAM : Action
- Éléments de la politique JSON IAM : NotResource
- Éléments de politique JSON IAM : Resource

Avertissement de sécurité : transférer le rôle avec étoile dans la ressource

Code d'erreur : PASS_ROLE_WITH_STAR_IN_RESOURCE

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Pass role with star in resource: Using the iam:PassRole action with wildcards (*)
 in the resource can be overly permissive because it allows iam:PassRole permissions
 on multiple resources. We recommend that you specify resource ARNs or add the
 iam:PassedToService condition key to your statement.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the iam:PassRole action with wildcards (*) in the resource can
 be overly permissive because it allows iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs or add the iam:PassedToService condition
 key to your statement."
```

Résolution de l'avertissement de sécurité

Pour configurer de nombreux AWS services, vous devez leur transmettre un rôle IAM. Pour permettre cela, vous devez accorder l'autorisation `iam:PassRole` à une identité (utilisateur, groupe d'utilisateurs ou rôle). Les politiques qui autorisent `iam:PassRole` et qui incluent un caractère générique (*) dans l'`Resource`élément peuvent permettre à vos mandants d'accéder à plus de services ou de fonctionnalités que vous ne le souhaitiez. AWS recommande de spécifier plutôt les ARN autorisés dans l'`Resource`élément. En outre, vous pouvez réduire les autorisations à un seul service en utilisant la clé de condition `iam:PassedToService`.

Certains AWS services incluent leur espace de noms de service dans le nom de leur rôle. Cette vérification de politique tient compte de ces conventions lors de l'analyse de la politique afin de générer des conclusions. Par exemple, l'ARN de ressource suivant peut ne pas générer de résultat :

```
arn:aws:iam::*:role/Service*
```

- Transférer un rôle à un service
- iam : PassedToService
- Éléments de politique JSON IAM : Resource

AWS politiques gérées avec cet avertissement de sécurité

AWS les politiques gérées vous permettent de commencer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

L'un de ces cas d'utilisation concerne les administrateurs de votre compte. Les politiques AWS gérées suivantes fournissent un accès administrateur et accordent des autorisations pour transmettre n'importe quel rôle IAM à n'importe quel service. AWS vous recommande d'associer les politiques AWS gérées suivantes uniquement aux identités IAM que vous considérez comme des administrateurs.

- AdministratorAccess-Amplify

Les politiques AWS gérées suivantes incluent des autorisations utilisant `iam:PassRole` un caractère générique (*) dans la ressource et sont sur une voie de dépréciation. Pour chacune de ces politiques, nous avons mis à jour les directives relatives aux autorisations, notamment en recommandant une nouvelle politique AWS gérée qui prend en charge le cas d'utilisation. Pour afficher des alternatives à ces politiques, veuillez consulter les guides de chaque service.

- AWSElasticBeanstalkFullAccess
- AWSElasticBeanstalkService
- AWSLambdaFullAccess
- AWSLambdaReadOnlyAccess
- AWSOpsWorksFullAccess
- AWSOpsWorksRole
- AWSDataPipelineRole
- AmazonDynamoDBFullAccesswithDataPipeline
- AmazonElasticMapReduceFullAccess
- AmazonDynamoDBFullAccesswithDataPipeline
- AmazonEC2ContainerServiceFullAccess

Les politiques AWS gérées suivantes fournissent des autorisations uniquement pour les rôles liés aux services, ce qui permet aux AWS services d'effectuer des actions en votre nom. Vous ne pouvez pas attacher ces politiques à vos identités IAM.

- AWSServiceRoleForAmazonEKSNodegroup

Avertissement de sécurité : transférer le rôle avec étoile dans l'action et la ressource

Code d'erreur : PASS_ROLE_WITH_STAR_IN_ACTION_AND_RESOURCE

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Pass role with star in action and resource: Using wildcards (*) in the action and the
 resource can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using wildcards (*) in the action and the resource can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
 that you specify resource ARNs or add the iam:PassedToService condition key to your
 statement."
```

Résolution de l'avertissement de sécurité

Pour configurer de nombreux AWS services, vous devez leur transmettre un rôle IAM. Pour permettre cela, vous devez accorder l'autorisation `iam:PassRole` à une identité (utilisateur, groupe d'utilisateurs ou rôle). Les politiques comportant un caractère générique (*) dans les `Resource` éléments `Action` et peuvent permettre à vos mandants d'accéder à un plus grand nombre de services ou de fonctionnalités que vous ne le souhaitiez. AWS recommande de spécifier plutôt les ARN autorisés dans l'`Resource`élément. En outre, vous pouvez réduire les autorisations à un seul service en utilisant la clé de condition `iam:PassedToService`.

- Transférer un rôle à un service
- iam : PassedToService
- Éléments de politique JSON IAM : Action
- Éléments de politique JSON IAM : Resource

AWS politiques gérées avec cet avertissement de sécurité

AWS les politiques gérées vous permettent de commencer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Certains de ces cas d'utilisation sont destinés aux administrateurs de votre compte. Les politiques AWS gérées suivantes fournissent un accès administrateur et accordent des autorisations pour transmettre n'importe quel rôle IAM à n'importe quel AWS service. AWS vous recommande d'associer les politiques AWS gérées suivantes uniquement aux identités IAM que vous considérez comme des administrateurs.

- AdministratorAccess
- IAMFullAccess

Avertissement de sécurité — Rôle de passe avec une étoile dans la ressource et NotAction

Code d'erreur : PASS_ROLE_WITH_STAR_IN_RESOURCE_AND_NOT_ACTION

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Pass role with star in resource and NotAction: Using a resource with wildcards (*) and
 NotAction can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
 that you specify resource ARNs or add the iam:PassedToService condition key to your
 statement."
```

Résolution de l'avertissement de sécurité

Pour configurer de nombreux AWS services, vous devez leur transmettre un rôle IAM. Pour permettre cela, vous devez accorder l'autorisation `iam:PassRole` à une identité (utilisateur, groupe d'utilisateurs ou rôle). L'utilisation de l'`NotAction`élément dans une politique avec un caractère générique (*) dans l'`Resource`élément peut permettre à vos mandants d'accéder à plus de services

ou de fonctionnalités que vous ne le souhaitiez. AWS recommande de spécifier plutôt les ARN autorisés dans l'`Resource`élément. En outre, vous pouvez réduire les autorisations à un seul service en utilisant la clé de condition `iam:PassedToService`.

- Transférer un rôle à un service
- iam : PassedToService
- Éléments de la politique JSON IAM : NotAction
- Éléments de politique JSON IAM : Action
- Éléments de politique JSON IAM : Resource

Avertissement de sécurité : clés de condition appariées manquantes

Code d'erreur : MISSING_PAIRED_CONDITION_KEYS

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing paired condition keys: Using the condition key {{conditionKeyName}}
 can be overly permissive without also using the following condition keys:
 {{recommendedKeys}}. Condition keys like this one are more secure when paired with
 a related key. We recommend that you add the related condition keys to the same
 condition block.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the condition key {{conditionKeyName}} can be overly
 permissive without also using the following condition keys: {{recommendedKeys}}.
 Condition keys like this one are more secure when paired with a related key. We
 recommend that you add the related condition keys to the same condition block."
```

Résolution de l'avertissement de sécurité

Certaines clés de condition sont plus sécurisées lorsqu'elles sont associées à d'autres clés de condition associées. AWS vous recommande d'inclure les clés de condition associées dans le même

bloc de conditions que la clé de condition existante. Cela rend les autorisations accordées via la politique plus sécurisées.

Par exemple, vous pouvez utiliser la clé de condition `aws:VpcSourceIp` pour comparer l'adresse IP à partir de laquelle une demande a été effectuée avec l'adresse IP que vous spécifiez dans la politique. AWS vous recommande d'ajouter la clé de condition `aws:SourceVPC` associée. Cela vérifie si la demande provient du VPC que vous spécifiez dans la politique et de l'adresse IP que vous spécifiez.

Termes connexes

- clé de condition globale `aws:VpcSourceIp`
- clé de condition globale `aws:SourceVPC`
- Clés de condition globale
- Élément de condition
- Présentation des politiques JSON

Avertissement de sécurité : refuser avec clé de condition de balise non prise en charge pour le service

Code d'erreur : DENY_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Deny with unsupported tag condition key for service: Using the effect Deny with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes can be overly permissive: {{serviceNames}}. Actions for the listed services
 are not denied by this statement. We recommend that you move these actions to a
 different statement without this condition key.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the effect Deny with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes can be
 overly permissive: {{serviceNames}}. Actions for the listed services are not denied
```

```
by this statement. We recommend that you move these actions to a different statement
without this condition key."
```

Résolution de l'avertissement de sécurité

L'utilisation de clés de condition de balise non prises en charge dans l'`Condition`élément d'une politique `"Effect": "Deny"` peut être trop permissive, car la condition est ignorée pour ce service. AWS vous recommande de supprimer les actions de service qui ne prennent pas en charge la clé de condition et de créer une autre instruction pour refuser l'accès à des ressources spécifiques pour ces actions.

Si vous utilisez la clé de condition `aws:ResourceTag` et qu'elle n'est pas prise en charge par une action de service, la clé n'est pas incluse dans le contexte de la demande. Dans ce cas, la condition dans l'instruction `Deny` renvoie toujours `false` et l'action n'est jamais refusée. Cela se produit même si la ressource est correctement balisée.

Lorsqu'un service prend en charge la clé de condition `aws:ResourceTag`, vous pouvez utiliser des balises pour contrôler l'accès aux ressources de ce service. Ceci est connu sous le nom de contrôle d'accès basé sur les attributs (ABAC). Les services qui ne prennent pas en charge ces clés ont besoin que vous contrôliez l'accès aux ressources à l'aide du Contrôle d'accès basé sur les ressources (RBAC).

> **Note**
>
> Certains services autorisent la prise en charge de la clé de condition `aws:ResourceTag` pour un sous-ensemble de leurs ressources et actions. IAM Access Analyzer renvoie les résultats des actions de service qui ne sont pas prises en charge. Par exemple, Amazon S3 prend en charge `aws:ResourceTag` pour un sous-ensemble de ses ressources. Pour afficher tous les types de ressources disponibles dans Amazon S3 qui prennent en charge la clé de condition `aws:ResourceTag`, veuillez consulter Types de ressources définis par Amazon S3 dans la Référence des autorisations de service.

Par exemple, supposons que vous souhaitiez refuser l'accès pour supprimer les balises de ressources spécifiques qui sont balisées avec la paire clé-valeur `status=Confidential`. Supposons également que cela vous AWS Lambda permet de baliser et de débaliser des ressources, mais qu'il ne prend pas en charge la clé de `aws:ResourceTag` condition. Pour refuser les actions de suppression pour AWS App Mesh et AWS Backup si cette balise est présente, utilisez la clé de `aws:ResourceTag` condition. Pour Lambda, utilisez une convention de dénomination des

ressources incluant le préfixe `"Confidential"`. Incluez ensuite une instruction séparée qui empêche la suppression des ressources avec cette convention de dénomination.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyDeleteSupported",
            "Effect": "Deny",
            "Action": [
                "appmesh:DeleteMesh",
                "backup:DeleteBackupPlan"
                ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/status": "Confidential"
                }
            }
        },
        {
            "Sid": "DenyDeleteUnsupported",
            "Effect": "Deny",
            "Action": "lambda:DeleteFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:status-Confidential*"
        }
    ]
}
```

> **Warning**
>
> N'utilisez pas la IfExists version... de l'opérateur de condition pour contourner cette découverte. Cela signifie « Supprimer l'action si la clé est présente dans le contexte de la demande et que les valeurs correspondent. Sinon, refuser l'action. » Dans l'exemple précédent, le fait d'inclure l'action `lambda:DeleteFunction` dans l'instruction `DenyDeleteSupported` avec l'opérateur `StringEqualsIfExists` refuse toujours l'action. Pour cette action, la clé n'est pas présente dans le contexte et chaque tentative de suppression de ce type de ressource est refusée, que celle-ci soit ou non balisée.

Termes connexes

- Clés de condition globale
- Comparaison entre ABAC et RBAC
- Éléments de politique JSON IAM : Opérateurs de condition
- Élément de condition
- Présentation des politiques JSON

Avertissement de sécurité — Refus NotAction avec clé de condition d'étiquette non prise en charge pour le service

Code d'erreur :
DENY_NOTACTION_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Deny NotAction with unsupported tag condition key for service: Using the effect Deny
 with NotAction and the tag condition key {{conditionKeyName}} can be overly permissive
 because some service actions are not denied by this statement. This is because the
 condition key doesn't apply to some service actions. We recommend that you use Action
 instead of NotAction.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the effect Deny with NotAction and the tag condition key
 {{conditionKeyName}} can be overly permissive because some service actions are not
 denied by this statement. This is because the condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction."
```

Résolution de l'avertissement de sécurité

L'utilisation de clés de condition de balise dans l'élément `Condition` d'une politique avec les éléments `NotAction` et `"Effect": "Deny"` peut être trop permissive. La condition est ignorée pour les actions de service qui ne prennent pas en charge la clé de condition. AWS vous recommande de réécrire la logique pour refuser une liste d'actions.

Si vous utilisez la clé de condition `aws:ResourceTag` avec `NotAction`, toutes les actions de service nouvelles ou existantes ne prenant pas en charge la clé ne sont pas refusées. AWS vous recommande de répertorier explicitement les actions que vous voulez refuser. IAM Access Analyzer renvoie un résultat distinct pour les actions répertoriées ne prenant pas en charge la clé de condition `aws:ResourceTag`. Pour plus d'informations, veuillez consulter Avertissement de sécurité : refuser avec clé de condition de balise non prise en charge pour le service.

Lorsqu'un service prend en charge la clé de condition `aws:ResourceTag`, vous pouvez utiliser des balises pour contrôler l'accès aux ressources de ce service. Ceci est connu sous le nom de contrôle d'accès basé sur les attributs (ABAC). Les services qui ne prennent pas en charge ces clés ont besoin que vous contrôliez l'accès aux ressources à l'aide du Contrôle d'accès basé sur les ressources (RBAC).

Termes connexes

- Clés de condition globale
- Comparaison entre ABAC et RBAC
- Éléments de politique JSON IAM : Opérateurs de condition
- Élément de condition
- Présentation des politiques JSON

Avertissement de sécurité : restreindre l'accès au principal de service

Code d'erreur : RESTRICT_ACCESS_TO_SERVICE_PRINCIPAL

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Restrict access to service principal: Granting access to a service principal without
 specifying a source is overly permissive. Use aws:SourceArn, aws:SourceAccount,
 aws:SourceOrgID, or aws:SourceOrgPaths condition key to grant fine-grained access.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Granting access to a service principal without specifying a source
 is overly permissive. Use aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or
 aws:SourceOrgPaths condition key to grant fine-grained access."
```

Résolution de l'avertissement de sécurité

Vous pouvez spécifier Services AWS dans l'`Principal`élément d'une politique basée sur les ressources à l'aide d'un principal de service, qui est un identifiant pour le service. Lorsque vous accordez l'accès à un principal de service pour agir en votre nom, limitez l'accès. Vous pouvez empêcher les politiques trop permissives en utilisant les touches`aws:SourceArn`, `aws:SourceAccount``aws:SourceOrgID`, ou `aws:SourceOrgPaths` condition pour restreindre l'accès à une source spécifique, telle qu'un ARN de ressource, Compte AWS un ID d'organisation ou des chemins d'organisation spécifiques. La restriction de l'accès permet d'éviter un problème de sécurité appelé le Problème de l'adjoint confus.

Termes connexes

- Service AWS directeurs
- AWS clés de condition globales : aws : SourceAccount
- AWS clés de condition globales : aws : SourceArn
- AWS clés de condition globales : aws : SourceOrgId
- AWS clés de condition globales : aws : SourceOrgPaths
- Le problème de l'adjoint confus

Avertissement de sécurité : clé de condition pour oidc principal manquante

Code d'erreur : MISSING_CONDITION_KEY_FOR_OIDC_PRINCIPAL

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing condition key for oidc principal: Using an Open ID Connect principal without a
 condition can be overly permissive. Add condition keys with a prefix that matches your
 federated OIDC principals to ensure that only the intended identity provider assumes
 the role.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using an Open ID Connect principal without a condition can be
 overly permissive. Add condition keys with a prefix that matches your federated OIDC
 principals to ensure that only the intended identity provider assumes the role."
```

Résolution de l'avertissement de sécurité

L'utilisation d'un principal Open ID Connect sans condition peut être trop permissive. Ajoutez des clés de condition avec un préfixe qui correspond à vos principaux OIDC fédérés pour garantir que seul le fournisseur d'identité prévu assume le rôle.

Termes connexes

- Création d'un rôle pour l'identité Web ou pour OpenID Connect Federation (console)

Avertissement de sécurité : clé de condition github repo manquante

Code d'erreur : MISSING_GITHUB_REPO_CONDITION_KEY

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Missing github repo condition key: Granting a federated GitHub principal permissions
 without a condition key can allow more sources to assume the role than you intended.
 Add the token.actions.githubusercontent.com:sub condition key and specify the branch
 and repository name in the value.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Granting a federated GitHub principal permissions without a
 condition key can allow more sources to assume the role than you intended. Add the
 token.actions.githubusercontent.com:sub condition key and specify the branch and
 repository name in the value."
```

Résolution de l'avertissement de sécurité

Si vous l'utilisez GitHub en tant qu'IdP OIDC, la meilleure pratique consiste à limiter le nombre d'entités pouvant assumer le rôle associé à l'IdP IAM. Lorsque vous incluez une `Condition` déclaration dans une politique de confiance relative aux rôles, vous pouvez limiter le rôle à une GitHub organisation, un référentiel ou une branche spécifique. Vous pouvez utiliser la clé de condition `token.actions.githubusercontent.com:sub` pour limiter l'accès. Nous vous recommandons de limiter la condition à un ensemble spécifique de référentiels ou de branches. Si vous n'incluez pas cette condition, GitHub les actions émanant d'organisations ou de référentiels indépendants de votre volonté peuvent assumer les rôles associés à l'IdP GitHub IAM de votre compte. AWS

Termes connexes

- Configuration d'un rôle pour le fournisseur d'identité GitHub OIDC

Avertissement de sécurité : opérateur semblable à une chaîne avec des clés de condition ARN

Code d'erreur : STRING_LIKE_OPERATOR_WITH_ARN_CONDITION_KEYS

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
String like operator with ARN condition keys: Use the operator type {{allowed}} instead
 of operator {{operator}} for the condition key {{key}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Use the operator type {{allowed}} instead of operator {{operator}}
 for the condition key {{key}}."
```

Résolution de l'avertissement de sécurité

AWS recommande d'utiliser des opérateurs ARN plutôt que des opérateurs de chaîne lors de la comparaison des ARN afin de garantir une restriction d'accès appropriée en fonction des valeurs

des conditions d'ARN. Mettez à jour l'opérateur `StringLike` en fonction de l'opérateur `ArnLike` de votre élément `Condition` chaque fois que la clé spécifiée est utilisée.

Ces politiques AWS gérées constituent des exceptions à cet avertissement de sécurité :

- AmazonSecurityLakeAdministrator
- AWSCodePipeline_FullAccess
- AWSCodePipeline_ReadOnlyAccess
- S3UnlockBucketPolicy
- SQSUnlockQueuePolicy

Termes connexes

- Opérateurs de condition Amazon Resource Name (ARN)
- Opérateurs de condition de chaîne
- AWS politiques gérées

Avertissement de sécurité — Attributs DynamoDB sans sélection

Code de problème : DYNAMODB_ATTRIBUTES_WITHOUT_SELECT

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
DynamoDB attributes without select: Restricting dynamodb:Attributes without also
 setting dynamodb:Select to SPECIFIC_ATTRIBUTES allows all attributes to be returned
 on read requests that omit a projection expression. We recommend that you also set
 dynamodb:Select to SPECIFIC_ATTRIBUTES.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Restricting dynamodb:Attributes without also setting dynamodb:Select
 to SPECIFIC_ATTRIBUTES allows all attributes to be returned on read requests that
```

```
omit a projection expression. We recommend that you also set dynamodb:Select to
SPECIFIC_ATTRIBUTES."
```

Résolution de l'avertissement de sécurité

Ajoutez une `StringEquals` condition définie `dynamodb:Select` sur `SPECIFIC_ATTRIBUTES` dans la même instruction que la `dynamodb:Attributes` condition. DynamoDB évalue `dynamodb:Attributes` uniquement les demandes qui spécifient les attributs à renvoyer. Sans cette condition, une demande de lecture qui omet une expression de projection renvoie l'élément entier.

Termes connexes

- Utilisation des conditions de politique IAM pour un contrôle d'accès précis
- Éléments de politique JSON IAM : Opérateurs de condition

Avertissement de sécurité — Préfixe S3 dans un contexte négatif

Code de problème : S3_PREFIX_IN_NEGATIVE_CONTEXT

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
S3 prefix in negative context: Using the s3:prefix condition key in a Deny statement or
 with a negated string operator does not block listing the objects under the matching
 prefixes. A request that specifies a shorter prefix does not match the condition and
 can still list those objects. To restrict listing to specific prefixes, use s3:prefix
 with StringLike in an Allow statement, or use a Deny statement with StringNotLike.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the s3:prefix condition key in a Deny statement or with
 a negated string operator does not block listing the objects under the matching
 prefixes. A request that specifies a shorter prefix does not match the condition and
 can still list those objects. To restrict listing to specific prefixes, use s3:prefix
 with StringLike in an Allow statement, or use a Deny statement with StringNotLike."
```

Résolution de l'avertissement de sécurité

Pour limiter la liste à des préfixes spécifiques, utilisez l'un des modèles suivants :

- Autoriser avec StringLike : accordez `s3:ListBucket` avec une `StringLike` condition concernant la `s3:prefix` liste des préfixes que vous souhaitez autoriser. Les requêtes qui fournissent un préfixe différent ne sont pas couvertes par cette autorisation et ne peuvent pas répertorier ces objets.
- Refuser avec StringNotLike — Refuser `s3:ListBucket` avec une `StringNotLike` condition sur la `s3:prefix` liste des préfixes que vous souhaitez autoriser. Une demande qui omet le préfixe ou en fournit un plus court ne correspond pas à l'ensemble autorisé et est refusée.

Un `Deny` with `StringLike` ou un `Allow` with `StringNotLike` ne restreint pas efficacement l'accès car un appelant peut contourner la condition en spécifiant un préfixe plus court (par exemple, `priv` au lieu de`private/`) qui ne correspond pas à la valeur de la condition.

Termes connexes

- Exemples de clés conditionnelles Amazon S3
- Éléments de politique JSON IAM : Opérateurs de condition
- Éléments de politique IAM : condition

Avertissement de sécurité — ForAnyValue avec type de réclamation d'audience

Code d'erreur : FORANYVALUE_WITH_AUDIENCE_CLAIM_TYPE

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
ForAnyValue with audience claim type: Using ForAnyValue qualifier with the single-
valued condition key {{key}} can be overly permissive. We recommend that you remove
 ForAnyValue:.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using ForAnyValue qualifier with the single-valued condition key
 {{key}} can be overly permissive. We recommend that you remove ForAnyValue:."
```

Résolution de l'avertissement de sécurité

AWS recommande de ne pas utiliser l'opérateur `ForAnyValue` set avec des clés de condition à valeur unique. Utilisez des opérateurs d'ensemble uniquement avec des clés de condition à valeurs multiples. Supprimez l'opérateur d'ensemble `ForAnyValue`.

Termes connexes

- Single-valued par rapport aux clés de contexte à valeurs multiples
- Single-valued exemples de politiques clés contextuelles

Avertissement de sécurité — Attributs DynamoDB sans valeurs de retour

Code de problème : DYNAMODB_ATTRIBUTES_WITHOUT_RETURN_VALUES

Type de résultat : SECURITY_WARNING

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
DynamoDB attributes without return values: Restricting dynamodb:Attributes without
 limiting dynamodb:ReturnValues allows write operations to return the entire item,
 which can expose attributes outside the allowed list. We recommend that you restrict
 dynamodb:ReturnValues to NONE, UPDATED_OLD, or UPDATED_NEW.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Restricting dynamodb:Attributes without limiting
 dynamodb:ReturnValues allows write operations to return the entire item, which
 can expose attributes outside the allowed list. We recommend that you restrict
 dynamodb:ReturnValues to NONE, UPDATED_OLD, or UPDATED_NEW."
```

Résolution de l'avertissement de sécurité

Ajoutez une condition qui `dynamodb:ReturnValues` se limite à `NONEUPDATED_OLD`, ou `UPDATED_NEW` dans la même instruction que la `dynamodb:Attributes` condition. Sans cette

limite, une écriture qui définit `dynamodb:ReturnValues` `ALL_OLD` ou `ALL_NEW` renvoie l'intégralité de l'élément, y compris les attributs autres que ceux que vous avez autorisés.

Suggestion : action de tableau vide

Code d'erreur : EMPTY_ARRAY_ACTION

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Empty array action: This statement includes no actions and does not affect the policy.
 Specify actions.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "This statement includes no actions and does not affect the policy.
 Specify actions."
```

Résolution de la suggestion

Les instructions doivent inclure un élément `Action` ou `NotAction` comprenant un ensemble d'actions. Lorsque l'élément est vide, l'instruction de politique ne fournit aucune autorisation. Spécifiez les actions dans l'élément `Action`.

- Éléments de politique IAM JSON : Action

Suggestion : condition de tableau vide

Code d'erreur : EMPTY_ARRAY_CONDITION

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Empty array condition: There are no values for the condition key {{key}} and it does
 not affect the policy. Specify conditions.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "There are no values for the condition key {{key}} and it does not
 affect the policy. Specify conditions."
```

Résolution de la suggestion

La structure de l'élément `Condition` facultatif a besoin que vous utilisiez un opérateur de condition et une paire clé-valeur. Lorsque la valeur de la condition est vide, la condition renvoie `true` et l'instruction de politique ne fournit aucune autorisation. Spécifiez une valeur de condition.

- Éléments de politique JSON IAM : Condition

Suggestion — État du tableau vide ForAllValues

Code d'erreur : EMPTY_ARRAY_CONDITION_FORALLVALUES

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Empty array condition ForAllValues: The ForAllValues prefix with an empty condition key
 matches only if the key {{key}} is missing from the request context. To determine if
 the request context is empty, we recommend that you use the Null condition operator
 with the value of true instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The ForAllValues prefix with an empty condition key matches only
 if the key {{key}} is missing from the request context. To determine if the request
 context is empty, we recommend that you use the Null condition operator with the value
 of true instead."
```

Résolution de la suggestion

La structure de l'élément `Condition` a besoin que vous utilisiez un opérateur de condition et une paire clé-valeur. L'opération d'ensemble `ForAllValues` teste si la valeur de chaque membre de la requête est un sous-ensemble de la clé de condition.

Lorsque vous utilisez `ForAllValues` avec une clé de condition vide, la condition ne correspond que s'il n'y a pas de clés dans la demande. AWS vous recommande plutôt d'utiliser l'opérateur de condition `Null` si vous voulez tester si un contexte de demande est vide.

- Clés de contexte à valeurs multiples
- Opérateur de condition null
- Éléments de politique JSON IAM : Condition

Suggestion — État du tableau vide ForAnyValue

Code d'erreur : EMPTY_ARRAY_CONDITION_FORANYVALUE

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Empty array condition ForAnyValue: The ForAnyValue prefix with an empty condition key
 {{key}} never matches the request context and it does not affect the policy. Specify
 conditions.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The ForAnyValue prefix with an empty condition key {{key}} never
 matches the request context and it does not affect the policy. Specify conditions."
```

Résolution de la suggestion

La structure de l'élément `Condition` a besoin que vous utilisiez un opérateur de condition et une paire clé-valeur. L'opérateur d'ensemble `ForAnyValues` teste si au moins un membre de l'ensemble des valeurs de la demande correspond à au moins un membre de l'ensemble des valeurs de la clé de condition.

Lorsque vous utilisez `ForAnyValues` avec une clé de condition vide, la condition ne correspond jamais. Cela signifie que la déclaration n'a aucun effet sur la politique. AWS vous recommande de réécrire la condition.

- Clés de contexte à valeurs multiples
- Éléments de politique JSON IAM : Condition

Suggestion — État du tableau vide IfExists

Code d'erreur : EMPTY_ARRAY_CONDITION_IFEXISTS

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Empty array condition IfExists: The IfExists suffix with an empty condition key matches
 only if the key {{key}} is missing from the request context. To determine if the
 request context is empty, we recommend that you use the Null condition operator with
 the value of true instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The IfExists suffix with an empty condition key matches only if the
 key {{key}} is missing from the request context. To determine if the request context
 is empty, we recommend that you use the Null condition operator with the value of true
 instead."
```

Résolution de la suggestion

Le suffixe `...IfExists` modifie un opérateur de condition. Ceci signifie que si la clé de politique est présente dans le contexte de la demande, la clé doit être traitée comme spécifié dans la politique. Si la clé n'est pas présente, la condition évalue l'élément de condition comme vrai. »

Lorsque vous utilisez `...IfExists` avec une clé de condition vide, la condition ne correspond que s'il n'y a pas de clés dans la demande. AWS vous recommande plutôt d'utiliser l'opérateur de condition `Null` si vous voulez tester si un contexte de demande est vide.

- ... IfExists opérateurs conditionnels
- Éléments de politique JSON IAM : Condition

Suggestion : principal de tableau vide

Code d'erreur : EMPTY_ARRAY_PRINCIPAL

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Empty array principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

Résolution de la suggestion

Vous devez utiliser l'`NotPrincipal`élément `Principal` ou dans les politiques de confiance pour les rôles IAM et dans les politiques basées sur les ressources. Resource-based les politiques sont des politiques que vous intégrez directement dans une ressource.

Lorsque vous fournissez un tableau vide dans l'`Principal`élément d'une instruction, celle-ci n'a aucun effet sur la politique. AWS vous recommande de spécifier les personnes principales qui doivent avoir accès à la ressource.

- Éléments de politique JSON IAM : principal
- Éléments de la politique JSON IAM : NotPrincipal

Suggestion : ressource de tableau vide

Code d'erreur : EMPTY_ARRAY_RESOURCE

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Empty array resource: This statement includes no resources and does not affect the
 policy. Specify resources.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "This statement includes no resources and does not affect the policy.
 Specify resources."
```

Résolution de la suggestion

Les instructions doivent inclure un élément `Resource` ou `NotResource`.

Lorsque vous fournissez un tableau vide dans l'élément de ressource d'une instruction, celle-ci n'a aucun effet sur la politique. AWS vous recommande de spécifier des Amazon Resource Names (ARN) pour les ressources.

- Éléments de politique JSON IAM : Resource
- Éléments de la politique JSON IAM : NotResource

Suggestion : condition d'objet vide

Code d'erreur : EMPTY_OBJECT_CONDITION

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Empty object condition: This condition block is empty and it does not affect the
 policy. Specify conditions.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "This condition block is empty and it does not affect the policy.
 Specify conditions."
```

Résolution de la suggestion

La structure de l'élément `Condition` a besoin que vous utilisiez un opérateur de condition et une paire clé-valeur.

Lorsque vous fournissez un objet vide dans l'élément de condition d'une instruction, l'instruction n'a aucun effet sur la politique. Supprimez l'élément facultatif ou spécifiez des conditions.

- Éléments de politique JSON IAM : Condition

Suggestion : principal d'objet vide

Code d'erreur : EMPTY_OBJECT_PRINCIPAL

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Empty object principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

Résolution de la suggestion

Vous devez utiliser l'`NotPrincipal`élément `Principal` ou dans les politiques de confiance pour les rôles IAM et dans les politiques basées sur les ressources. Resource-based les politiques sont des politiques que vous intégrez directement dans une ressource.

Lorsque vous fournissez un objet vide dans l'`Principal`élément d'une instruction, celle-ci n'a aucun effet sur la politique. AWS vous recommande de spécifier les personnes principales qui doivent avoir accès à la ressource.

- Éléments de politique JSON IAM : principal
- Éléments de la politique JSON IAM : NotPrincipal

Suggestion : valeur sid vide

Code d'erreur : EMPTY_SID_VALUE

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Empty Sid value: Add a value to the empty string in the Sid element.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a value to the empty string in the Sid element."
```

Résolution de la suggestion

L'élément `Sid` (ID de instruction) facultatif est un identifiant que vous pouvez fournir pour l'instruction de politique. Vous pouvez affecter une valeur `Sid` à chaque instruction d'un tableau d'instructions. Si vous sélectionnez d'utiliser l'élément `Sid`, vous devez indiquer une valeur de chaîne.

Termes connexes

- Éléments de politique JSON IAM : Sid

Suggestion : équivaut à null false

Code d'erreur : EQUIVALENT_TO_NULL_FALSE

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Equivalent to null false: We recommend replacing the key {{key}} in the condition block
 of {{operator}} with {{{recommendedKey}}: false} to ensure better enforcement of the
 condition.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "We recommend replacing the key {{key}} in the condition block of
 {{operator}} with {{{recommendedKey}}: false} to ensure better enforcement of the
 condition."
```

Résolution de la suggestion

Remplacez la clé de condition actuelle par la clé recommandée définie sur `false`. Ce changement améliore la clarté de la politique et garantit une évaluation plus fiable des conditions. Mettez à jour votre bloc conditionnel afin d'utiliser `{recommendedKey}: false` à la place de la combinaison clé-opérateur actuelle.

Termes connexes

- Éléments de politique IAM : condition
- Conditions avec plusieurs clés ou valeurs de contexte
- AWS clés de contexte de condition globale

Suggestion : équivaut à null true

Code d'erreur : EQUIVALENT_TO_NULL_TRUE

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Equivalent to null true: We recommend replacing the key {{key}} in the condition block
 of {{operator}} with {{{recommendedKey}}: true} to ensure better enforcement of the
 condition.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "We recommend replacing the key {{key}} in the condition block of
 {{operator}} with {{{recommendedKey}}: true} to ensure better enforcement of the
 condition."
```

Résolution de la suggestion

Remplacez la clé de condition actuelle par la clé recommandée définie sur `true`. Ce changement améliore la clarté de la politique et garantit une évaluation plus fiable des conditions. Mettez à jour votre bloc conditionnel afin d'utiliser `{recommendedKey}: true` à la place de la combinaison clé-opérateur actuelle.

Termes connexes

- Éléments de politique IAM : condition
- Conditions avec plusieurs clés ou valeurs de contexte
- AWS clés de contexte de condition globale

Suggestion : améliorer une plage IP

Code d'erreur : IMPROVE_IP_RANGE

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Improve IP range: The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}."
```

Résolution de la suggestion

Les conditions d'adresse IP doivent être au format CIDR standard, tel que 203.0.113. 0/24 ou 2001 DB8:1234:5678 : :/64. Lorsque vous incluez des bits non nuls après les bits masqués, ils ne sont pas pris en compte pour la condition. AWS vous recommande d'utiliser la nouvelle adresse incluse dans le message.

- Opérateurs de condition d'adresse IP
- Éléments de politique JSON IAM : Condition

Suggestion : Null avec qualificateur

Code d'erreur : NULL_WITH_QUALIFIER

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Null with qualifier: Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively."
```

Résolution de la suggestion

Dans l'élément `Condition`, vous créez des expressions dans lesquelles vous utilisez des opérateurs de condition tels que égal ou inférieur à pour comparer une condition de la politique avec des clés et des valeurs dans le contexte de la demande. Pour les demandes incluant plusieurs valeurs pour une seule clé de condition, vous devez utiliser les opérateurs définis `ForAllValues` ou `ForAnyValue`.

Lorsque vous utilisez l'opérateur de condition `Null` avec `ForAllValues`, l'instruction renvoie toujours `true`. Lorsque vous utilisez l'opérateur de `Null` condition with`ForAnyValue`, l'instruction est toujours renvoyée`false`. AWS recommande d'utiliser l'opérateur de `StringLike` condition avec ces opérateurs définis.

Termes connexes

- Clés de contexte à valeurs multiples
- Opérateur de condition null
- Élément de condition

Suggestion : sous-ensemble d'adresses IP privées

Code d'erreur : PRIVATE_IP_ADDRESS_SUBSET

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Private IP address subset: The values for condition key aws:SourceIp include a mix
 of private and public IP addresses. The private addresses will not have the desired
 effect. aws:SourceIp works only for public IP address ranges. To define permissions
 for private IP ranges, use aws:VpcSourceIp.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses will not have the desired effect.
 aws:SourceIp works only for public IP address ranges. To define permissions for
 private IP ranges, use aws:VpcSourceIp."
```

Résolution de la suggestion

La clé de condition globale `aws:SourceIp` fonctionne uniquement pour les plages d'adresses IP publiques.

Lorsque votre élément `Condition` inclut un mélange d'adresses IP privées et publiques, l'instruction peut ne pas avoir l'effet désiré. Vous pouvez spécifier des adresses IP privées avec `aws:VpcSourceIP`.

> **Note**
>
> La clé de condition globale `aws:VpcSourceIP` ne correspond que si la demande provient de l'adresse IP spécifiée et qu'elle passe par un point de terminaison de VPC.

- aws : clé de condition SourceIp globale
- aws : clé de condition VpcSourceIp globale
- Opérateurs de condition d'adresse IP
- Éléments de politique JSON IAM : Condition

Suggestion — NotIpAddress Sous-ensemble privé

Code d'erreur : PRIVATE_NOT_IP_ADDRESS_SUBSET

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Private NotIpAddress subset: The values for condition key aws:SourceIp include a mix of
 private and public IP addresses. The private addresses have no effect. aws:SourceIp
 works only for public IP address ranges. To define permissions for private IP ranges,
 use aws:VpcSourceIp.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses have no effect. aws:SourceIp works
 only for public IP address ranges. To define permissions for private IP ranges, use
 aws:VpcSourceIp."
```

Résolution de la suggestion

La clé de condition globale `aws:SourceIp` fonctionne uniquement pour les plages d'adresses IP publiques.

Lorsque votre élément `Condition` inclut l'opérateur de condition `NotIpAddress` et un mélange d'adresses IP privées et publiques, l'instruction peut ne pas avoir l'effet désiré. Toutes les adresses IP publiques non spécifiées dans la politique correspondront. Aucune adresse IP privée ne correspondra. Pour obtenir cet effet, vous pouvez utiliser `NotIpAddress` avec `aws:VpcSourceIP` et spécifier les adresses IP privées qui ne doivent pas correspondre.

- aws : clé de condition SourceIp globale
- aws : clé de condition VpcSourceIp globale
- Opérateurs de condition d'adresse IP
- Éléments de politique JSON IAM : Condition

Suggestion : action redondante

Code d'erreur : REDUNDANT_ACTION

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Redundant action: The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}."
```

Résolution de la suggestion

Lorsque vous utilisez des caractères génériques (*) dans l'`Action`élément, vous pouvez inclure des autorisations redondantes. AWS vous recommande de revoir votre politique et de n'inclure que les autorisations dont vous avez besoin. Cela peut vous aider à supprimer les actions redondantes.

Par exemple, les actions suivantes incluent l'action `iam:GetCredentialReport` deux fois.

```
"Action": [
        "iam:Get*",
        "iam:List*",
        "iam:GetCredentialReport"
```

```
    ],
```

Dans cet exemple, les autorisations sont définies pour chaque action IAM commençant par `Get` ou `List`. Lorsque IAM ajoute des opérations get ou list supplémentaires, cette politique les autorise. Vous pouvez vouloir autoriser toutes ces actions en lecture seule. L'action `iam:GetCredentialReport` est déjà incluse dans le cadre de `iam:Get*`. Pour supprimer les autorisations en double, vous pouvez supprimer `iam:GetCredentialReport`.

Vous recevez un résultat pour cette vérification de politique lorsque tout le contenu d'une action est redondant. Dans cet exemple, si l'élément incluait `iam:*CredentialReport`, il ne serait pas considéré comme redondant. En effet, il inclut `iam:GetCredentialReport`, qui est redondant, et `iam:GenerateCredentialReport`, qui ne l'est pas. La suppression de `iam:Get*` ou `iam:*CredentialReport` modifierait les autorisations de la politique.

- Éléments de politique JSON IAM : Action

AWS politiques gérées avec cette suggestion

AWS les politiques gérées vous permettent de commencer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Les actions redondantes n'affectent pas les autorisations accordées par la politique. Lorsque vous utilisez une politique AWS gérée comme référence pour créer votre politique gérée par le client, vous AWS recommande de supprimer les actions redondantes de votre politique.

Suggestion : valeur de condition redondante num

Code d'erreur : REDUNDANT_CONDITION_VALUE_NUM

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Redundant condition value num: Multiple values in {{operator}} are redundant. Replace
 with the {{greatest/least}} single value for {{key}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Multiple values in {{operator}} are redundant. Replace with the
 {{greatest/least}} single value for {{key}}."
```

Résolution de la suggestion

Lorsque vous utilisez des opérateurs de condition numériques pour des valeurs similaires dans une clé de condition, vous pouvez créer un chevauchement qui entraîne des autorisations redondantes.

Par exemple, l'élément `Condition` suivant inclut plusieurs conditions `aws:MultiFactorAuthAge` dont les plages d'd'âge se chevauchent de 1 200 secondes.

```
"Condition": {
        "NumericLessThan": {
          "aws:MultiFactorAuthAge": [
            "2700",
            "3600"
          ]
        }
      }
```

Dans cet exemple, les autorisations sont définies si l'authentification multifacteur (MFA) a été effectuée il y a moins de 3 600 secondes (1 heure). Vous pouvez supprimer la valeur `2700` redondante.

- Opérateurs de condition numériques
- Éléments de politique JSON IAM : Condition

Suggestion : ressource redondante

Code d'erreur : REDUNDANT_RESOURCE

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Redundant resource: The {{redundantResourceCount}} resource ARN(s) are redundant
 because they reference the same resource. Review the use of wildcards (*)
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{redundantResourceCount}} resource ARN(s) are redundant because
 they reference the same resource. Review the use of wildcards (*)"
```

Résolution de la suggestion

Lorsque vous utilisez des caractères génériques (*) dans les Amazon Resource Names (ARN), vous pouvez créer des autorisations de ressource redondante.

Par exemple, l'élément `Resource` suivant inclut plusieurs ARN avec des autorisations redondantes.

```
"Resource": [
            "arn:aws:iam::111122223333:role/jane-admin",
            "arn:aws:iam::111122223333:role/jane-s3only",
            "arn:aws:iam::111122223333:role/jane*"
        ],
```

Dans cet exemple, les autorisations sont définies pour tous les rôles dont le nom commence par `jane`. Vous pouvez supprimer les ARN `jane-admin` et `jane-s3only` redondants sans modifier les autorisations qui en résultent. Cela rend la politique dynamique. Elle définira les autorisations pour tous les futurs rôles commençant par `jane`. Si la politique vise à autoriser l'accès à un nombre statique de rôles, supprimez le dernier ARN et répertoriez uniquement les ARN qui doivent être définis.

- Éléments de politique JSON IAM : Resource

AWS politiques gérées avec cette suggestion

AWS les politiques gérées vous permettent de commencer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Les ressources redondantes n'affectent pas les autorisations accordées par la politique. Lorsque vous utilisez une politique AWS gérée comme référence pour créer votre politique gérée par le client, vous AWS recommande de supprimer les ressources redondantes de votre politique.

Suggestion : instruction redondante

Code d'erreur : REDUNDANT_STATEMENT

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Redundant statement: The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement."
```

Résolution de la suggestion

L'élément `Statement` constitue l'élément principal d'une politique. Cet élément est obligatoire. L'élément `Statement` peut contenir une seule instruction ou un ensemble d'instructions individuelles.

Lorsque vous incluez la même instruction plusieurs fois dans une politique longue, les instructions sont redondantes. Vous pouvez supprimer l'une des instructions sans affecter les autorisations accordées par la politique. Lorsqu'une personne modifie une politique, elle peut modifier l'une des instructions sans mettre à jour son double. Cela peut entraîner un nombre d'autorisations plus élevé que prévu.

- Éléments de politique JSON IAM : Statement

Suggestion : caractère générique dans le nom du service

Code d'erreur : WILDCARD_IN_SERVICE_NAME

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Wildcard in service name: Avoid using wildcards (*, ?) in the service name because it
 might grant unintended access to other AWS services with similar names.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Avoid using wildcards (*, ?) in the service name because it might
 grant unintended access to other AWS services with similar names."
```

Résolution de la suggestion

Lorsque vous incluez le nom d'un AWS service dans une politique, AWS recommande de ne pas inclure de caractères génériques (*, ?). Cela peut ajouter des autorisations non voulues à de futurs services. Par exemple, il existe plus d'une douzaine de AWS services dont le nom `*code*` contient le mot.

```
"Resource": "arn:aws:*code*::111122223333:*"
```

- Éléments de politique JSON IAM : Resource

Suggestion : autoriser avec la clé de condition d'étiquette non prise en charge pour le service

Code d'erreur : ALLOW_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Allow with unsupported tag condition key for service: Using the effect Allow with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes does not affect the policy: {{serviceNames}}. Actions for the listed service
 are not allowed by this statement. We recommend that you move these actions to a
 different statement without this condition key.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the effect Allow with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes does not
```

```
affect the policy: {{serviceNames}}. Actions for the listed service are not allowed
by this statement. We recommend that you move these actions to a different statement
without this condition key."
```

Résolution de la suggestion

L'utilisation de clés de condition de balise non prises en charge dans l'`Condition`élément d'une politique avec n'`"Effect": "Allow"`affecte pas les autorisations accordées par la stratégie, car la condition est ignorée pour cette action de service. AWS recommande de supprimer les actions pour les services qui ne prennent pas en charge la clé de condition et de créer une autre instruction pour autoriser l'accès à des ressources spécifiques de ce service.

Si vous utilisez la clé de condition `aws:ResourceTag` et qu'elle n'est pas prise en charge par une action de service, la clé n'est pas incluse dans le contexte de la demande. Dans ce cas, la condition dans l'instruction `Allow` renvoie toujours `false` et l'action n'est jamais autorisée. Cela se produit même si la ressource est correctement balisée.

Lorsqu'un service prend en charge la clé de condition `aws:ResourceTag`, vous pouvez utiliser des balises pour contrôler l'accès aux ressources de ce service. Ceci est connu sous le nom de contrôle d'accès basé sur les attributs (ABAC). Les services qui ne prennent pas en charge ces clés ont besoin que vous contrôliez l'accès aux ressources à l'aide du Contrôle d'accès basé sur les ressources (RBAC).

> **Note**
>
> Certains services autorisent la prise en charge de la clé de condition `aws:ResourceTag` pour un sous-ensemble de leurs ressources et actions. IAM Access Analyzer renvoie les résultats des actions de service qui ne sont pas prises en charge. Par exemple, Amazon S3 prend en charge `aws:ResourceTag` pour un sous-ensemble de ses ressources. Pour afficher tous les types de ressources disponibles dans Amazon S3 qui prennent en charge la clé de condition `aws:ResourceTag`, veuillez consulter Types de ressources définis par Amazon S3 dans la Référence des autorisations de service.

Par exemple, supposons que vous vouliez autoriser des membres de l'équipe à afficher les détails des ressources spécifiques qui sont balisées avec la paire clé-valeur `team=BumbleBee`. Supposons également que cela vous AWS Lambda permet de baliser des ressources, mais qu'il ne prend pas en charge la clé de `aws:ResourceTag` condition. Pour autoriser les actions d'affichage pour AWS App Mesh et AWS Backup si cette balise est présente, utilisez la clé de `aws:ResourceTag` condition.

Pour Lambda, utilisez une convention de dénomination des ressources incluant le nom de l'équipe comme préfixe. Incluez ensuite une instruction séparée qui permet l'affichage des ressources avec cette convention de dénomination.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewSupported",
            "Effect": "Allow",
            "Action": [
                "appmesh:DescribeMesh",
                "backup:GetBackupPlan"
                ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/team": "BumbleBee"
                }
            }
        },
        {
            "Sid": "AllowViewUnsupported",
            "Effect": "Allow",
            "Action": "lambda:GetFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:team-BumbleBee*"
        }
    ]
}
```

Warning

N'utilisez pas la version de l'opérateur de condition `Not` avec `"Effect": "Allow"` pour contourner ce résultat. Ces opérateurs de condition fournissent une correspondance annulée. Cela signifie qu'une fois la condition évaluée, le résultat est annulé. Dans l'exemple précédent, le fait d'inclure l'action `lambda:GetFunction` dans l'instruction `AllowViewSupported` avec l'opérateur `StringNotEquals` autorise toujours l'action, que la ressource soit balisée ou non.
N'utilisez pas la `IfExists` version... de l'opérateur de condition pour contourner cette découverte. Cela signifie « Autoriser l'action si la clé est présente dans le contexte de la demande et que les valeurs correspondent. Sinon, autoriser l'action. » Dans

l'exemple précédent, le fait d'inclure l'action `lambda:GetFunction` dans l'instruction `AllowViewSupported` avec l'opérateur `StringEqualsIfExists` autorise toujours l'action. Pour cette action, la clé n'est pas présente dans le contexte et chaque tentative d'affichage de ce type de ressource est autorisée, que la ressource soit ou non étiquetée.

Termes connexes

- Clés de condition globale
- Éléments de politique JSON IAM : Opérateurs de condition
- Élément de condition
- Présentation des politiques JSON

Suggestion — Autoriser NotAction avec une clé de condition de balise non prise en charge pour le service

Code d'erreur :
ALLOW_NOTACTION_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Allow NotAction with unsupported tag condition key for service: Using the effect Allow
 with NotAction and the tag condition key {{conditionKeyName}} allows only service
 actions that support the condition key. The condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the effect Allow with NotAction and the tag condition key
 {{conditionKeyName}} allows only service actions that support the condition key. The
 condition key doesn't apply to some service actions. We recommend that you use Action
 instead of NotAction."
```

Résolution de la suggestion

L'utilisation de clés de condition de balise non prises en charge dans l'élément `Condition` d'une politique avec l'élément `NotAction` et `"Effect": "Allow"` n'affecte pas les autorisations accordées par la politique. La condition est ignorée pour les actions de service qui ne prennent pas en charge la clé de condition. AWS recommande de réécrire la logique pour autoriser une liste d'actions.

Si vous utilisez la clé de condition `aws:ResourceTag` avec `NotAction`, toutes les actions de service nouvelles ou existantes ne prenant pas en charge la clé ne sont pas autorisées. AWS vous recommande de répertorier explicitement les actions que vous voulez autoriser. IAM Access Analyzer renvoie un résultat distinct pour les actions répertoriées ne prenant pas en charge la clé de condition `aws:ResourceTag`. Pour plus d'informations, veuillez consulter Suggestion : autoriser avec la clé de condition d'étiquette non prise en charge pour le service.

Lorsqu'un service prend en charge la clé de condition `aws:ResourceTag`, vous pouvez utiliser des balises pour contrôler l'accès aux ressources de ce service. Ceci est connu sous le nom de contrôle d'accès basé sur les attributs (ABAC). Les services qui ne prennent pas en charge ces clés ont besoin que vous contrôliez l'accès aux ressources à l'aide du Contrôle d'accès basé sur les ressources (RBAC).

Termes connexes

- Clés de condition globale
- Comparaison entre ABAC et RBAC
- Éléments de politique JSON IAM : Opérateurs de condition
- Élément de condition
- Présentation des politiques JSON

Suggestion : clé de condition recommandée pour le principal de service

Code d'erreur : RECOMMENDED_CONDITION_KEY_FOR_SERVICE_PRINCIPAL

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Recommended condition key for service principal: To restrict access to the service
 principal {{servicePrincipalPrefix}} operating on your behalf, we recommend
 aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of
 {{key}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "To restrict access to the service principal
 {{servicePrincipalPrefix}} operating on your behalf, we recommend aws:SourceArn,
 aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of {{key}}."
```

Résolution de la suggestion

Vous pouvez spécifier Services AWS dans l'`Principal`élément d'une politique basée sur les ressources à l'aide d'un principal de service, qui est un identifiant pour le service. Vous devez utiliser les clés de condition `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` ou `aws:SourceOrgPaths` lors de l'octroi d'un accès à des principaux de service plutôt qu'à d'autres clés de condition, telles que `aws:Referer`. Cela vous aide à prévenir un problème de sécurité appelé le problème de l'adjoint confus.

Termes connexes

- Service AWS directeurs
- AWS clés de condition globales : aws : SourceAccount
- AWS clés de condition globales : aws : SourceArn
- AWS clés de condition globales : aws : SourceOrgId
- AWS clés de condition globales : aws : SourceOrgPaths
- Le problème de l'adjoint confus

Suggestion : clé de condition non pertinente dans la politique

Code d'erreur : IRRELEVANT_CONDITION_KEY_IN_POLICY

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Irrelevant condition key in policy: The condition key {{condition-key}} is not relevant
 for the {{resource-type}} policy.  Use this key in an identity-based policy to govern
 access to this resource.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key {{condition-key}} is not relevant for the
 {{resource-type}} policy.  Use this key in an identity-based policy to govern access
 to this resource."
```

Résolution de la suggestion

Certaines clés de condition ne sont pas pertinentes pour les politiques basées sur les ressources. Par exemple, la clé de condition `s3:ResourceAccount` n'est pas pertinente pour la politique basée sur les ressources attachée à un compartiment Amazon S3 ou à un type de ressource de point d'accès Amazon S3.

Vous devez utiliser la clé de condition dans une politique basée sur l'identité pour contrôler l'accès à la ressource.

Termes connexes

- Identity-based politiques et politiques fondées sur les ressources

Suggestion : clé redondante en raison du caractère générique dans la condition

Code d'erreur : REDUNDANT_KEY_DUE_TO_WILDCARD_IN_CONDITION

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Redundant key due to wildcard in condition: The key {{key}} in the condition block of
 {{operator}} is redundant because it is always matched. Remove this key to simplify
 the condition.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The key {{key}} in the condition block of {{operator}} is redundant
 because it is always matched. Remove this key to simplify the condition."
```

Résolution de la suggestion

Supprimez la clé de condition redondante de votre politique. La clé correspond toujours en raison du motif de caractères génériques, ce qui la rend inutile. Simplifiez votre bloc de conditions en supprimant cette clé tout en conservant les mêmes autorisations effectives.

Termes connexes

- Éléments de politique IAM : condition
- Conditions avec plusieurs clés ou valeurs de contexte
- AWS clés de contexte de condition globale

Suggestion : principal redondant dans la politique d'approbation de rôle

Code d'erreur : REDUNDANT_PRINCIPAL_IN_ROLE_TRUST_POLICY

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Redundant principal in role trust policy: The assumed-role principal
 {{redundant_principal}} is redundant with its parent role {{parent_role}}. Remove the
 assumed-role principal.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The assumed-role principal {{redundant_principal}} is redundant with
 its parent role {{parent_role}}. Remove the assumed-role principal."
```

Résolution de la suggestion

Si vous spécifiez à la fois un principal avec rôle assumé et son rôle parent dans l'élément `Principal` d'une politique, celle-ci n'autorise ou ne refuse aucune autorisation différente. Par exemple, il est redondant si vous spécifiez l'élément `Principal` au format suivant :

```
"Principal": {
            "AWS": [
            "arn:aws:iam::AWS-account-ID:role/rolename",
            "arn:aws:iam::AWS-account-ID:assumed-role/rolename/rolesessionname"
        ]
```

Nous recommandons de supprimer le principal avec rôle assumé.

Termes connexes

- Principaux de séance de rôle

Suggestion : instruction redondante en raison du caractère générique dans la condition

Code d'erreur : REDUNDANT_STATEMENT_DUE_TO_WILDCARD_IN_CONDITION

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Redundant statement due to wildcard in condition: The key {{key}} in the condition
 block of {{operator}} does not match any values. Remove this key to simplify the
 condition.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The key {{key}} in the condition block of {{operator}} does not
 match any values. Remove this key to simplify the condition."
```

Résolution de la suggestion

Supprimez la clé de condition qui ne correspond à aucune valeur. Cette clé crée une condition inaccessible qui ne sera jamais satisfaite, la rendant ainsi redondante. Nettoyez votre politique en supprimant cette clé pour améliorer la lisibilité et les performances.

Termes connexes

- Éléments de politique IAM : condition
- Conditions avec plusieurs clés ou valeurs de contexte
- AWS clés de contexte de condition globale

Suggestion : confirmer le type de réclamation de l'audience

Code d'erreur : CONFIRM_AUDIENCE_CLAIM_TYPE

Type de résultat : SUGGESTION

Trouver des détails

Dans le Console de gestion AWS, les résultats de cette vérification incluent le message suivant :

```
Confirm audience claim type: The "{{key}}" ({{audienceType}}) claim key identifies
 the recipients that the JSON web token is intended for. Because this claim is single-
valued, do not use a qualifier.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The "{{key}}" ({{audienceType}}) claim key identifies the recipients
 that the JSON web token is intended for. Because this claim is single-valued, do not
 use a qualifier."
```

Résolution de la suggestion

La clé de réclamation (d'audience) `aud` est un identifiant unique pour votre application qui vous est délivré lorsque vous enregistrez votre application auprès de l'IdP, et elle identifie les destinataires auxquels le jeton web JSON est destiné. Les reclamations d'audience peuvent être à valeurs multiples ou à valeur unique. Si la réclamation est à valeurs multiples, utilisez un opérateur d'ensemble de conditions `ForAllValues` ou `ForAnyValue`. Si la réclamation est à valeur unique, n'utilisez pas d'opérateur d'ensemble de conditions.

Termes connexes

- Création d'un rôle pour l'identité Web ou pour OpenID Connect Federation (console)

- Clés de contexte à valeurs multiples
- Single-valued par rapport aux clés de condition à valeurs multiples

Validation des politiques à l'aide des vérifications des politiques personnalisées de l'analyseur d'accès IAM

Vous pouvez utiliser des vérifications de politique personnalisées pour vérifier les nouveaux accès en fonction de vos normes de sécurité. Des frais sont associés à chaque vérification pour un nouvel accès. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

Validation des politiques à l'aide de vérifications de politique personnalisées (console)

En option, vous pouvez exécuter une vérification de politique personnalisées lorsque vous modifiez une politique dans l'éditeur de politique JSON dans la console IAM. Vous pouvez vérifier si la politique mise à jour accorde un nouvel accès par rapport à la version existante.

Pour vérifier les nouveaux accès lors de la modification des politiques IAM JSON

1. Connectez-vous à la console IAM Console de gestion AWS et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de gauche, sélectionnez Policies (Politiques).
3. Dans la liste des politiques, choisissez le nom de la politique à modifier. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Choisissez l'onglet Autorisations, puis Modifier.
5. Choisissez l'option JSON et mettez à jour votre politique.
6. Dans le panneau de validation de la politique ci-dessous, sélectionnez l'onglet Vérifier un nouvel accès, puis sélectionnez Vérifier la politique. Si les autorisations modifiées accordent un nouvel accès, la déclaration sera mise en évidence dans le volet de validation de la politique.
7. Si vous n'avez pas l'intention d'accorder un nouvel accès, mettez à jour la déclaration de politique et choisissez Vérifier la politique jusqu'à ce qu'aucun nouvel accès ne soit détecté.

> **Note**
>
> Des frais sont associés à chaque vérification pour un nouvel accès. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

8. Choisissez Suivant.
9. Sur la page examiner et enregistrer, examinez les Autorisations définies dans cette politique, puis choisissez Enregistrer les modifications.

Validation des politiques à l'aide de vérifications de politiques personnalisées (AWS CLI ou API)

Vous pouvez exécuter des vérifications de politique personnalisées par IAM Access Analyzer à partir de l'API AWS CLI ou de l'API d'IAM Access Analyzer.

Pour exécuter les vérifications de politiques personnalisées de l'analyseur d'accès IAM (AWS CLI)

- Pour vérifier si un nouvel accès est autorisé pour une politique mise à jour par rapport à la politique existante, exécutez la commande suivante : check-no-new-access
- Pour vérifier si l'accès spécifié n'est pas autorisé par une politique, exécutez la commande suivante : check-access-not-granted
- Pour vérifier si une politique de ressources peut accorder un accès public à un type de ressource spécifique, exécutez la commande suivante : check-no-public-access

Pour exécuter les vérifications de politiques personnalisées de l'analyseur d'accès IAM (API)

- Pour vérifier si un nouvel accès est autorisé pour une politique mise à jour par rapport à la politique existante, utilisez l'opération d'API CheckNoNewAccess.
- Pour vérifier si l'accès spécifié n'est pas autorisé par une politique, utilisez l'opération d'API CheckAccessNotGranted.
- Pour vérifier si une politique de ressources peut accorder un accès public à un type de ressource spécifique, utilisez l'opération d'API CheckNoPublicAccess.

Génération d'une politique IAM Access Analyzer

En tant qu'administrateur ou développeur, vous pouvez octroyer plus d'autorisations à des entités IAM (utilisateurs ou rôles) qu'elle n'en ont besoin. IAM propose plusieurs options pour vous aider à affiner les autorisations que vous octroyez.

Une option consiste à générer une politique IAM basée sur une activité d'accès pour une entité. IAM Access Analyzer examine vos AWS CloudTrail journaux et génère un modèle de politique contenant

les autorisations utilisées par l'entité au cours de la plage de dates que vous avez spécifiée. Vous pouvez utiliser le modèle pour créer une politique avec des autorisations précises qui accordent uniquement les autorisations requises pour prendre en charge votre cas d'utilisation spécifique.

Rubriques

- Processus de génération de politique
- Informations de niveau service et action
- Points à noter sur la génération de politiques
- Autorisations nécessaires pour générer une politique
- Générer une politique en fonction de CloudTrail l'activité (console)
- Générez une politique à l'aide de AWS CloudTrail données d'un autre compte
- Générez une politique basée sur CloudTrail l'activité (AWS CLI)
- Générez une politique basée sur CloudTrail l'activité (AWS API)
- Services de génération d'une politique d'Analyseur d'accès IAM

Processus de génération de politique

IAM Access Analyzer analyse vos CloudTrail événements pour identifier les actions et les services qui ont été utilisés par une entité IAM (utilisateur ou rôle). Il génère ensuite une politique IAM basée sur cette activité. Vous pouvez ajuster les autorisations d'une entité en remplaçant la politique d'autorisations étendues attachée à l'entité par la politique générée. Voici un aperçu général du processus de génération de politique.

- Configuration de la génération de modèles de politique : vous spécifiez une période pouvant aller jusqu'à 90 jours pour qu'IAM Access Analyzer analyse vos événements historiques AWS CloudTrail . Vous devez spécifier un rôle de service existant ou en créer un nouveau. Le rôle de service permet à IAM Access Analyzer d'accéder à vos informations de CloudTrail suivi et de dernier accès au service afin d'identifier les services et les actions qui ont été utilisés. Vous devez spécifier le CloudTrail journal qui enregistre les événements pour le compte avant de pouvoir générer une politique. Pour plus d'informations sur les quotas de CloudTrail données d'IAM Access Analyzer, consultez la section Quotas d'IAM Access Analyzer.
- Générer une politique : IAM Access Analyzer génère une politique en fonction de l'activité d'accès dans vos CloudTrail événements.
- Examiner et personnaliser la politique : Une fois la politique générée, vous pouvez consulter les services et actions qui ont été utilisés par l'entité pendant la plage de dates spécifiée. Vous pouvez

personnaliser davantage la politique en ajoutant ou en supprimant des autorisations, en spécifiant des ressources et en ajoutant des conditions au modèle de politique.

- Créer et attacher une politique : Vous avez la possibilité d'enregistrer la politique générée en créant une politique gérée. Vous pouvez attacher la politique que vous créez à l'utilisateur ou au rôle dont l'activité a été utilisée pour générer la politique.

Informations de niveau service et action

Lorsque IAM Access Analyzer génère une politique IAM, des informations sont renvoyées pour vous aider à la personnaliser. Deux catégories d'informations peuvent être renvoyées lorsqu'une politique est générée :

- Politique contenant des informations relatives aux actions : pour certains AWS services, tels qu'Amazon EC2, IAM Access Analyzer peut identifier les actions détectées dans vos CloudTrail événements et répertorie les actions utilisées dans la politique qu'il génère. Pour obtenir la liste des services pris en charge, consultez Services de génération d'une politique d'Analyseur d'accès IAM. Pour certains services, IAM Access Analyzer vous invite à ajouter des actions pour les services à la politique générée.
- Politique avec des informations de niveau service : IAM Access Analyzer utilise les informations relatives aux derniers services consultés pour créer un modèle de politique avec tous les services récemment utilisés. Lorsque vous utilisez le Console de gestion AWS, nous vous invitons à consulter les services et à ajouter des actions pour compléter la politique.

Pour obtenir la liste des actions de chaque service, consultez la section Actions, ressources et clés de condition pour les AWS services dans la référence d'autorisation des services.

Points à noter sur la génération de politiques

Avant de générer une politique, veuillez consulter les conseils clés suivants.

- Activer un CloudTrail suivi : un suivi doit être activé CloudTrail pour que votre compte puisse générer une politique basée sur l'activité d'accès. Lorsque vous créez un CloudTrail suivi, CloudTrail envoie les événements liés à votre parcours vers un compartiment Amazon S3 que vous spécifiez. Pour savoir comment créer un CloudTrail parcours, consultez la rubrique Création d'un parcours pour votre AWS compte dans le Guide de AWS CloudTrail l'utilisateur.

- Événements de données non disponibles : l'analyseur d'accès IAM n'identifie pas l'activité au niveau action pour les événements de données, tels que les événements de données Amazon S3, dans les politiques générées.
- PassRole— L'`iam:PassRole`action n'est pas suivie CloudTrail et n'est pas incluse dans les politiques générées.
- Réduire le temps de génération des politiques– Pour générer une politique plus rapidement, réduisez la plage de dates spécifiée lors de la configuration.
- Utilisation CloudTrail à des fins d'audit : n'utilisez pas la génération de politiques à des fins d'audit ; utilisez-la à la CloudTrail place. Pour plus d'informations sur l'utilisation CloudTrail, consultez la section Journalisation des appels IAM et AWS STS API avec AWS CloudTrail.
- Actions refusées : la génération de politiques passe en revue tous les CloudTrail événements, y compris les actions refusées.
- Une politique par console IAM : Vous pouvez générer une politique à la fois dans la console IAM.
- Disponibilité des politiques générées dans la console IAM : Vous pouvez consulter une politique générée dans la console IAM pendant 7 jours au maximum après sa génération. Après 7 jours, vous devez générer une nouvelle politique.
- Quotas de génération de politiques : pour plus d'informations sur les quotas de génération de politiques d'analyseur d'accès IAM, veuillez consulter Quotas IAM Access Analyzer.
- Les tarifs standard Amazon S3 s'appliquent : lorsque vous utilisez la fonctionnalité de génération de politiques, IAM Access Analyzer examine CloudTrail les journaux de votre compartiment S3. Il n'y a pas de frais de stockage supplémentaires pour accéder à vos CloudTrail journaux à des fins de génération de politiques. AWS facture les tarifs Amazon S3 standard pour les demandes et le transfert de données des CloudTrail journaux stockés dans votre compartiment S3.
- AWS Control Tower support — La génération de politiques ne prend pas en charge l'utilisation des AWS CloudTrail pistes AWS Control Tower créées lors de la génération de politiques, pour les raisons suivantes :
  - Les CloudTrail données de l'organisation sont enregistrées dans un autre compte, le compte AWS Control Tower Log Archive.
  - Les autorisations pour le compartiment S3 où ces journaux sont stockés ne peuvent pas être reconfigurées en raison des restrictions imposées au compartiment de journalisation S3 par les politiques de contrôle des services (SCP) d' AWS Control Tower.

Autorisations nécessaires pour générer une politique

Les autorisations dont vous avez besoin pour générer une politique pour la première fois diffèrent de celles dont vous avez besoin pour générer une politique les fois suivantes. Pour de plus amples informations, veuillez consulter Démarrage avec AWS Identity and Access Management Access Analyzer.

First-time configuration

Lorsque vous générez une politique pour la première fois, vous devez choisir un rôle de service existant approprié dans votre compte ou créer un nouveau rôle de service. Le rôle de service permet à IAM Access Analyzer d'accéder aux dernières informations consultées sur votre compte CloudTrail et de gérer les dernières informations consultées. Seuls les administrateurs doivent disposer des autorisations nécessaires pour créer et configurer des rôles. Par conséquent, nous recommandons qu'un administrateur crée le rôle de service lors de la première installation. Pour en savoir plus sur les autorisations requises pour créer des rôles de service, voir Création d'un rôle pour déléguer des autorisations à un AWS service.

Autorisations nécessaires pour le rôle de service

Lorsque vous créez un rôle de service, vous configurez deux politiques pour ce rôle. Vous attachez une politique d'autorisations IAM au rôle qui spécifie ce que le rôle peut faire. Vous lui attachez également une politique d'approbation de rôle, qui spécifie le principal qui peut utiliser le rôle.

Le premier exemple de politique correspond à une politique d'autorisations à attacher au rôle de service requis pour générer une politique. Le deuxième exemple de politique correspond à une politique d'approbation de rôle à attacher au rôle de service. Vous pouvez utiliser ces politiques pour vous aider à créer un rôle de service lorsque vous utilisez l' AWS API ou AWS CLI pour générer une politique. Lorsque vous utilisez la console IAM pour créer un rôle de service dans le cadre du processus de génération de politique, nous générons ces politiques pour vous.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "cloudtrail:GetTrail",
            "Resource": "*"
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetServiceLastAccessedDetails",
                "iam:GenerateServiceLastAccessedDetails"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

L'exemple de politique suivant correspond à une politique d'approbation de rôle avec les autorisations permettant à IAM Access Analyzer d'endosser le rôle.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "access-analyzer.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Utilisations ultérieures

Pour générer des politiques dans le Console de gestion AWS, un utilisateur IAM doit disposer d'une politique d'autorisations lui permettant de transmettre le rôle de service utilisé pour la génération de politiques à IAM Access Analyzer. `iam:PassRole`est généralement accompagné `iam:GetRole` pour que l'utilisateur puisse obtenir les détails du rôle à transmettre. Dans cet exemple, l'utilisateur peut transférer uniquement les rôles qui existent dans le compte spécifié avec des noms qui commencent par `AccessAnalyzerMonitorServiceRole*`. Pour en savoir plus sur la transmission de rôles IAM aux AWS services, consultez la section Octroi à un utilisateur des autorisations lui permettant de transmettre un rôle à un AWS service.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowUserToPassRole",
      "Effect": "Allow",
      "Action": [
        "iam:GetRole",
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::123456789012:role/service-role/
AccessAnalyzerMonitorServiceRole*"
    }
  ]
}
```

Vous devez également disposer des autorisations IAM Access Analyzer suivantes pour générer des politiques dans l' AWS API Console de gestion AWS, ou AWS CLI comme indiqué dans la déclaration de politique suivante.

```
{
  "Sid": "AllowUserToGeneratePolicy",
  "Effect": "Allow",
  "Action": [
    "access-analyzer:CancelPolicyGeneration",
    "access-analyzer:GetGeneratedPolicy",
    "access-analyzer:ListPolicyGenerations",
```

```
    "access-analyzer:StartPolicyGeneration"
  ],
  "Resource": "*"
}
```

Pour la première fois et les utilisations ultérieures

Lorsque vous utilisez le Console de gestion AWS pour générer une politique, vous devez être `cloudtrail:ListTrails` autorisé à répertorier les CloudTrail parcours sur votre compte, comme indiqué dans la déclaration de politique suivante.

```
{
  "Sid": "AllowUserToListTrails",
  "Effect": "Allow",
  "Action": [
    "cloudtrail:ListTrails"
  ],
  "Resource": "*"
}
```

Générer une politique en fonction de CloudTrail l'activité (console)

Vous pouvez générer une politique pour un utilisateur ou un rôle IAM.

Étape 1 : Générez une politique basée sur CloudTrail l'activité

La procédure suivante explique comment générer une politique pour un rôle à l'aide de la Console de gestion AWS.

Générer une politique pour un rôle IAM

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation de gauche, sélectionnez Roles (Rôles).

> **Note**
>
> Les étapes à suivre pour générer une politique basée sur l'activité d'un utilisateur IAM sont presque identiques. Dans ce cas, sélectionnez Users (Utilisateurs) au lieu de Roles (Rôles).

3. Dans la liste des rôles de votre compte, sélectionnez le nom du rôle dont vous souhaitez utiliser l'activité pour générer une politique.
4. Dans l'onglet Autorisations, dans la section Générer une politique en fonction CloudTrail des événements, choisissez Générer une politique.
5. Sur la page Générer une politique, spécifiez la période pendant laquelle vous souhaitez qu'IAM Access Analyzer analyse vos CloudTrail événements pour déterminer les actions entreprises avec le rôle. Vous pouvez choisir une période allant jusqu'à 90 jours. Nous vous recommandons de choisir la période la plus courte possible afin de réduire le temps de génération de politique.
6. Dans la section CloudTrail Accès, choisissez un rôle existant approprié ou créez-en un nouveau s'il n'existe pas de rôle approprié. Le rôle donne à IAM Access Analyzer les autorisations nécessaires pour accéder à vos CloudTrail données en votre nom afin d'examiner les activités d'accès afin d'identifier les services et les actions qui ont été utilisés. Pour en savoir plus sur les autorisations requises pour ce rôle, veuillez consulter Autorisations nécessaires pour générer une politique.
7. Dans la section CloudTrail trail to be analyzed (Piste Cloudtrail à analyser), spécifiez la piste CloudTrail qui consigne les événements pour le compte.

   Si vous choisissez une CloudTrail piste qui stocke les journaux d'un autre compte, une boîte d'informations sur l'accès entre comptes s'affiche. Cross-accountl'accès nécessite une configuration supplémentaire. Pour en savoir plus, veuillez consulter Choose a role for cross-account access plus loin dans cette rubrique.
8. Sélectionnez Generate policy (Générer une politique).
9. Lorsque la génération de politique est en cours, vous revenez à la page Roles Summary (Récapitulatif des rôles) sous l'onglet Permissions (Autorisations). Attendez que le statut de la section Policy request details (Détails de la demande de politique) affiche Success (Succès), puis sélectionnez View generated policy (Afficher la politique générée). Vous pouvez afficher la politique générée pendant sept jours au maximum. Si vous générez une autre politique, la politique existante est remplacée par la nouvelle politique que vous générez.

Étape 2 : Examiner les autorisations et ajouter des actions pour les services utilisés

Examinez les services et les mesures que l'analyseur d'accès IAM a constaté être utilisés par le rôle. Vous pouvez ajouter des actions pour tous les services qui ont été utilisés dans le modèle de politique généré.

1. Examinez les sections suivantes :

- Sur la page Review permissions (Vérifier les autorisations), veuillez consulter la liste des Actions included in the generated policy (Actions incluses dans la politique générée). La liste affiche les services et actions que l'analyseur d'accès IAM a constaté être utilisés par le rôle dans la plage de dates spécifiée.
- La section Services used (Services utilisés) affiche les services supplémentaires que l'analyseur d'accès IAM a constaté être utilisés par le rôle dans la plage de dates spécifiée. Il est possible que les informations sur les actions utilisées ne soient pas disponibles pour les services répertoriés dans cette section. Utilisez les menus de chaque service répertorié pour choisir manuellement les actions à inclure dans la politique.

2. Lorsque vous avez terminé d'ajouter des actions, cliquez sur Next (Suivant).

Étape 3 : Personnaliser la politique générée

Vous pouvez personnaliser la politique en ajoutant ou en supprimant des autorisations ou en spécifiant des ressources.

Pour personnaliser la politique générée

1. Mettez à jour le modèle de politique. Le modèle de politique contient des espaces réservés d'ARN de ressource pour les actions qui prennent en charge les autorisations au niveau des ressources, comme illustré dans l'image suivante. Resource-level les autorisations font référence à la capacité de spécifier les ressources sur lesquelles les utilisateurs sont autorisés à effectuer des actions. Nous vous recommandons d'utiliser des ARN pour spécifier vos ressources individuelles dans la politique pour les actions qui prennent en charge les autorisations au niveau des ressources. Vous pouvez remplacer les ARN de ressource d'espace réservé par des ARN de ressources valides pour votre cas d'utilisation.

   Si une action ne prend pas en charge les autorisations au niveau des ressources, utilisez un caractère générique (*) pour spécifier que toutes les ressources peuvent être concernées par l'action. Pour savoir quels AWS services prennent en charge les autorisations au niveau des ressources, consultez la section AWS Services qui fonctionnent avec IAM. Pour obtenir la liste des actions dans chaque service et pour savoir quelles actions prennent en charge les autorisations au niveau des ressources, veuillez consulter Actions, ressources et clés de condition pour les services AWS.



2. (Facultatif) Ajoutez, modifiez ou supprimez des instructions de politique JSON dans le modèle. Pour en savoir plus sur l'écriture de politiques JSON, veuillez consulter Creating IAM policies (Création de politiques IAM) (console).
3. Lorsque vous avez terminé de personnaliser le modèle de politique, vous disposez des options suivantes :

   - (Facultatif) Vous pouvez copier le JSON dans le modèle pour l'utiliser séparément en dehors de la page Generated policy (Politique générée). Par exemple, si vous souhaitez utiliser le JSON pour créer une politique dans un autre compte. Si la politique de votre modèle dépasse la limite de 6 144 caractères pour les politiques JSON, la politique est divisée en plusieurs politiques.
   - Cliquez sur Next (Suivant) pour consulter et créer une politique gérée dans le même compte.

Étape 4 : Examiner et créer une politique gérée

Si vous disposez des autorisations nécessaires pour créer et attacher des politiques IAM, vous pouvez créer une politique gérée à partir de la politique qui a été générée. Vous pouvez ensuite attacher la politique à un utilisateur ou à un rôle de votre compte.

Pour examiner et créer une politique

1. Dans la page Review and create managed policy (Examiner et créer une politique), spécifiez un nom sous Name et une Description (facultatif) pour la politique que vous êtes en train de créer.
2. (Facultatif) Dans la section Summary (Résumé), vous pouvez consulter les autorisations qui seront incluses dans la politique.
3. (Facultatif) Ajoutez des métadonnées à la politique en associant les balises sous forme de paires clé-valeur. Pour de plus amples informations sur l'utilisation de balises dans IAM, veuillez consulter Balisage des ressources IAM.
4. Lorsque vous avez terminé, effectuez l'une des opérations suivantes :
   - Vous pouvez attacher la nouvelle politique directement au rôle qui a été utilisé pour générer la politique. Pour ce faire, en bas de la page, cochez la case à côté de la politique Joindre à ***YourRoleName***. Ensuite, sélectionnez Create and attach policy (Créer et attacher une politique).
   - Sinon, cliquez sur Create policy (Créer une politique). Vous trouverez la politique que vous avez créée dans la liste des politiques du panneau de navigation Policies (Politiques) de la console IAM.
5. Vous pouvez attacher la politique que vous avez créée à une entité de votre compte. Après avoir attaché la politique, vous pouvez supprimer toute autre politique trop étendue qui pourrait être attachée à l'entité. Pour savoir comment attacher une politique gérée, veuillez consulter Adding IAM identity permissions (Ajout d'autorisations d'identité IAM) (console).

Générez une politique à l'aide de AWS CloudTrail données d'un autre compte

Vous pouvez créer des CloudTrail traces qui stockent les données dans des comptes centraux afin de rationaliser les activités de gouvernance. Par exemple, vous pouvez l'utiliser AWS Organizations pour créer un journal qui enregistre tous les événements pour tous les membres Comptes AWS de cette organisation. La piste appartient à un compte central.

Si vous souhaitez générer une politique pour un utilisateur ou un rôle dans un compte différent de celui sur lequel vos données de CloudTrail journal sont stockées, vous devez autoriser l'accès entre comptes. Pour ce faire, vous avez besoin à la fois d'un rôle et d'une politique de compartiment qui accordent des autorisations IAM Access Analyzer à vos CloudTrail journaux. Pour plus d'informations

sur la création de pistes Organisations, veuillez consulter Creating a trail for an organization (Création d'une piste pour une organisation).

Dans cet exemple, supposons que vous souhaitiez générer une politique pour un utilisateur ou un rôle dans le compte A. Le CloudTrail suivi du compte A stocke les CloudTrail journaux dans un compartiment du compte B. Avant de pouvoir générer une politique, vous devez effectuer les mises à jour suivantes :

1. Choisissez un rôle existant ou créez un nouveau rôle de service qui permet à IAM Access Analyzer d'accéder au compartiment du compte B (où sont stockés vos CloudTrail journaux).
2. Vérifiez votre politique de propriété d'objets de compartiment Amazon S3 et d'autorisations de compartiment dans le compte B afin que l'analyseur d'accès IAM puisse accéder aux objets du compartiment.

Étape 1 : choisir ou créer un rôle pour l'accès intercompte

- Sur l'écran Generate policy (Générer une politique), l'optionUse an existing role (Utiliser un rôle existant) est présélectionnée pour vous si un rôle avec les autorisations requises existe dans votre compte. Sinon, sélectionnez Create and use a new service role (Créer et utiliser un nouveau rôle de service). Le nouveau rôle est utilisé pour accorder à IAM Access Analyzer l'accès à vos CloudTrail journaux dans le compte B.

Étape 2 : vérifier la configuration du compartiment Amazon S3 dans le compte B

1. Connectez-vous à Console de gestion AWS et ouvrez la console Amazon S3 à l'adresse https://console.aws.amazon.com/s3/.
2. Dans la liste des compartiments, choisissez le nom du compartiment dans lequel vos journaux de CloudTrail randonnée sont stockés.
3. Choisissez l'onglet Permissions (Autorisations) et accédez à la section Object ownership (Propriété des objets).

   Utilisez les paramètres de propriété des objets du compartiment Amazon S3 pour contrôler la propriété des objets que vous chargez dans vos compartiments. Par défaut, lorsque d'autres Comptes AWS objets sont chargés dans votre compartiment, le compte de téléchargement est propriétaire des objets. Pour générer une politique, le propriétaire du compartiment doit posséder tous les objets du compartiment. Selon votre cas d'utilisation ACL, vous pouvez modifier le

réglage Object Ownership (Propriété de l'objet) pour votre compartiment. Définissez Object Ownership (Propriété de l'objet) sur l'une des options suivantes.

- Bucket owner enforced (Propriétaire du compartiment imposé) (recommandé)
- Bucket owner preferred (Propriétaire du compartiment préféré)

> **Important**
>
> Pour générer une politique avec succès, les objets du compartiment doivent appartenir au propriétaire du compartiment. Si vous choisissez d'utiliser Bucket owner preferred (Propriétaire du compartiment préféré), vous ne pouvez générer une politique que pour la période qui suit la modification de propriété de l'objet.

Pour en savoir plus sur la propriété des objets dans Amazon S3, veuillez consulter la rubrique Contrôle de la propriété des objets et désactivation des ACL pour votre compartiment dans le Guide de l'utilisateur d'Simple Storage Service (Amazon S3).

4. Ajoutez des autorisations à votre politique de compartiment Amazon S3 dans le compte B afin d'autoriser l'accès au rôle dans le compte A.

   L'exemple de politique suivant autorise `ListBucket` et `GetObject` pour le compartiment nommé `amzn-s3-demo-bucket`. Il autorise l'accès si le rôle accédant au compartiment appartient à un compte de votre organisation et a un nom commençant par `AccessAnalyzerMonitorServiceRole`. L'utilisation d'`aws:PrincipalArn` en tant que `Condition` dans l'élément `Resource` garantit que le rôle ne peut accéder à l'activité du compte que s'il appartient au compte A. Vous pouvez remplacer `amzn-s3-demo-bucket` par le nom de votre compartiment, `optional-prefix` par un préfixe facultatif pour le compartiment et `organization-id` par votre ID d'organisation.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PolicyGenerationBucketPolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
```

```
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket",
        "arn:aws:s3:::amzn-s3-demo-bucket/optional-prefix/AWSLogs/organization-id/
${aws:PrincipalAccount}/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "organization-id"
        },
        "ArnLike": {
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
      }
    }
  ]
}
```

5. Si vous chiffrez vos journaux à l'aide de AWS KMS, mettez à jour votre politique de AWS KMS clé dans le compte dans lequel vous stockez les CloudTrail journaux afin d'autoriser IAM Access Analyzer à utiliser votre clé, comme indiqué dans l'exemple de politique suivant. Remplacez `CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN` par l'ARN pour votre piste et `organization-id` par l'ID de votre organisation.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "kms:Decrypt",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:cloudtrail:arn":
 "CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN",
          "aws:PrincipalOrgID": "organization-id"
```

```
        },
        "StringLike": {
          "kms:ViaService": [
            "access-analyzer.*.amazonaws.com",
            "s3.*.amazonaws.com"
            ]
        },
        "ArnLike": {
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
      }
     }
    ]
}
```

Générez une politique basée sur CloudTrail l'activité (AWS CLI)

Vous pouvez utiliser les commandes suivantes pour générer une politique à l'aide de la AWS CLI.

Pour générer une politique

- aws accessanalyzer start-policy-generation

Pour afficher une politique générée

- aws accessanalyzer get-generated-policy

Pour annuler une demande de génération de politique

- aws accessanalyzer cancel-policy-generation

Pour afficher la liste des demandes de génération de politique

- aws accessanalyzer list-policy-generations

Générez une politique basée sur CloudTrail l'activité (AWS API)

Vous pouvez utiliser les opérations suivantes pour générer une politique à l'aide de l' AWS API.

Pour générer une politique

- StartPolicyGeneration

Pour afficher une politique générée

- GetGeneratedPolicy

Pour annuler une demande de génération de politique

- CancelPolicyGeneration

Pour afficher la liste des demandes de génération de politique

- ListPolicyGenerations

Services de génération d'une politique d'Analyseur d'accès IAM

Le tableau suivant répertorie les AWS services pour lesquels IAM Access Analyzer génère des politiques avec des informations au niveau de l'action. Pour obtenir la liste des actions de chaque service, consultez la section Actions, ressources et clés de condition pour les AWS services dans la référence d'autorisation de service.

AWS fournit des informations de génération de politiques au format JSON afin de rationaliser l'automatisation des flux de travail de gestion des politiques. Grâce aux informations de référence du service, vous pouvez accéder aux informations de génération de politiques à Services AWS partir de fichiers lisibles par machine. Pour plus d'informations, consultez les Informations sur les Service AWS simplifiés pour l'accès programmatique dans la Référence des autorisations de service.

Service	Préfixe de service
AWS Identity and Access Management Access Analyzer	access-analyzer
Gestion de compte AWS	compte
AWS Certificate Manager	acm
Amazon Managed Workflows for Apache Airflow	airflow

Service	Préfixe de service
AWS Amplify	amplify
AWS Amplify Générateur d'interface utilisateur	amplifyuibuilder
Amazon AppIntegrations	app-integrations
AWS AppConfig	appconfig
Amazon AppFlow	appflow
AWS Profileur des coûts d'application	application-cost-profiler
Informations sur les CloudWatch applications Amazon	applicationinsights
AWS App Mesh	appmesh
WorkSpaces Applications Amazon	appstream
AWS AppSync	appsync
Amazon Managed Service for Prometheus	aps
Amazon Athena	athéna
AWS Audit Manager	auditmanager
AWS Auto Scaling	autoscaling
AWS Marketplace	aws-marketplace
AWS Backup	sauvegarde
AWS Batch	lot
Amazon Braket	braket
AWS Budgets	Budgets
AWS Cloud9	Cloud9

Service	Préfixe de service
AWS CloudFormation	cloudformation
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm
Amazon CloudSearch	cloudsearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch
AWS CodeArtifact	codeartifact
AWS CodeDeploy	codedeploy
Amazon CodeGuru Profiler	codeguru-profiler
CodeGuru Réviseur Amazon	codeguru-reviewer
AWS CodePipeline	codepipeline
AWS CodeStar	codestar
AWS CodeStar Notifications	codestar-notifications
Amazon Cognito Identity	cognito-identity
Groupes d'utilisateurs Amazon Cognito	cognito-idp
Amazon Cognito Sync	cognito-sync
Amazon Comprehend Medical	comprehendmedical
Optimiseur de calcul AWS	compute-optimizer
AWS Config	config

Service	Préfixe de service
Amazon Connect Customer	connexion
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	dataexchange
AWS Data Pipeline	datapipeline
DynamoDB Accelerator	dax
AWS Device Farm	devicefarm
Amazon DevOps Guru	devops-guru
AWS Direct Connect	directconnect
Amazon Data Lifecycle Manager	dlm
AWS Database Migration Service	dms
Clusters élastiques Amazon DocumentDB	docdb-elastic
Amazon DynamoDB	dynamodb
Amazon Elastic Block Store	ebs
Amazon Elastic Compute Cloud	ec2
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Public	ecr-public
Amazon Elastic Container Service	ecs
Amazon Elastic Kubernetes Service	eks
Amazon ElastiCache	elasticache

Service	Préfixe de service
AWS Elastic Beanstalk	elasticbeanstalk
Amazon Elastic File System	elasticfilesystem
Elastic Load Balancing	elasticloadbalancing
Amazon EMR sur EKS (conteneurs EMR)	emr-containers
Amazon EMR sans serveur	emr-serverless
Amazon OpenSearch Service	es
Amazon EventBridge	events
Amazon FinSpace	finspace
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms
Amazon Fraud Detector	frauddetector
Amazon FSx	fsx
GameLift Serveurs Amazon	gamelift
Amazon Location Service	geo
Amazon Glacier	glacier
Amazon Managed Grafana	grafana
AWS IoT Greengrass	greengrass
AWS Ground Station	groundstation
Amazon GuardDuty	guardduty

Service	Préfixe de service
AWS HealthLake	healthlake
Amazon Honeycode	honeycode
Gestion des identités et des accès AWS	iam
AWS Boutique d'identités	identitystore
EC2 Image Builder	imagebuilder
Amazon Inspector Classic	inspector
Amazon Inspector	inspector2
AWS IoT	iot
AWS IoT Analytics	iotanalytics
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents
AWS IoT Fleet Hub	iotfleethub
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	iottwinmaker
AWS IoT Wireless	iotwireless
Amazon Interactive Video Service	ivs
Service de vidéo interactive Amazon Chat	ivschat
Streaming géré par Amazon pour Apache Kafka	kafka
Amazon Managed Streaming for Kafka Connect	kafkaconnect
Amazon Kendra	kendra

Service	Préfixe de service
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalytics
AWS Key Management Service	kms
AWS Lambda	lambda
Amazon Lex V2	lex
AWS License Manager Gestionnaire des abonnements Linux	license-manager-linux-subscriptions
Amazon Lightsail	lightsail
Amazon CloudWatch Logs	journaux
Amazon Lookout for Equipment	lookoutequipment
Amazon Lookout for Metrics	lookoutmetrics
AWS Mainframe Modernization	m2
Amazon Managed Blockchain	managedblockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB	base de données de mémoire
AWS Transform MGN	mgn
AWS Migration Hub	mgh

Service	Préfixe de service
AWS Migration Hub Strategy Recommendations	migration hub-strategy
Amazon Pinpoint	mobiletargeting
Amazon MQ	mq
AWS Network Manager	networkmanager
Amazon Nimble Studio	nimble
AWS HealthOmics	omics
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	outposts
AWS Organizations	organisations
AWS Performance Insights	pi
Amazon EventBridge Pipes	pipes
Amazon Polly	polly
Profils des clients Amazon Connect	profile
Amazon QLDB	qldb
AWS Resource Access Manager	ram
AWS Corbeille	rbin
Amazon Relational Database Service	rds
Amazon Redshift	redshift
API de données Amazon Redshift	redshift-data

Service	Préfixe de service
AWS Migration Hub Refactor Spaces	refactor-spaces
Amazon Rekognition	rekognition
AWS Resilience Hub	resiliencehub
Explorateur de ressources AWS	resource-explorer-2
Groupes de ressources AWS	resource-groups
AWS RoboMaker	robomaker
Gestion des identités et des accès AWS Des rôles partout	rolesanywhere
Amazon Route 53	route53
Amazon Route 53 Recovery Controls	route53-recovery-control-config
Amazon Route 53 Recovery Readiness	route53-recovery-readiness
Amazon Route 53 Resolver	route53resolver
AWS CloudWatch RUM	rum
Amazon Simple Storage Service	s3
Amazon S3 sur Outposts	s3-outposts
Fonctionnalités géospatiales d'Amazon SageMaker AI	sagemaker-geospatial
Savings Plans	savingsplans
EventBridge Schémas Amazon	schemas
Amazon SimpleDB	sdb

Service	Préfixe de service
AWS Secrets Manager	secretsmanager
AWS Security Hub CSPM	securityhub
Amazon Security Lake	securitylake
AWS Serverless Application Repository	serverlessrepo
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas
Amazon Simple Email Service	ses
AWS Shield	shield
AWS Signer	signer
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
Service de messages SMS et vocaux Amazon Pinpoint	sms-voice
AWS Snowball Edge	snowball
Amazon Simple Queue Service	sqs
AWS Systems Manager	ssm
AWS Systems Manager Incident Manager	ssm-incidents
Gestionnaire de systèmes AWS pour SAP	ssm-sap
AWS Step Functions	states
AWS Security Token Service	sts

Service	Préfixe de service
Amazon Simple Workflow Service	swf
Amazon CloudWatch Synthetics	synthetics
AWS Resource Groups Tagging API	balise
Amazon Textract	textract
Amazon Timestream	timestream
AWS Générateur de réseaux de télécommunications	tnb
Amazon Transcribe	transcribe
AWS Transfer Family	transfert
Amazon Translate	translate
Connect Customer Voice ID	voiceid
Amazon VPC Lattice	vpc-lattice
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Le savoir des clients d'Amazon Connect	wisdom
Amazon WorkLink	worklink
Amazon WorkSpaces	espaces de travail
AWS X-Ray	xray

Informations relatives aux actions de génération de politique

Le tableau suivant répertorie les actions pour lesquels l'analyseur d'accès IAM génère des politiques avec des informations de niveau action.

> **Important**
>
> L'action `iam:UpdateAccountName` deviendra obsolète le 22 avril 2026. Après le 22 avril 2026, seule l'autorisation `account:PutAccountName` contrôlera l'accès à la mise à jour du nom du compte. Nous vous recommandons vivement de mettre à jour toutes les politiques de contrôle des services (SCP) qui contrôlent les mises à jour des noms de compte pour utiliser l'autorisation `account:PutAccountName`.

Préfixe de service	Actions
access-analyzer	analyseur d'accès : ApplyArchiveRule
	analyseur d'accès : CancelPolicyGeneration
	analyseur d'accès : CheckAccessNotGranted
	analyseur d'accès : CheckNoNewAccess
	analyseur d'accès : CheckNoPublicAccess
	analyseur d'accès : CreateAccessPreview
	analyseur d'accès : CreateAnalyzer
	analyseur d'accès : CreateArchiveRule
	analyseur d'accès : DeleteAnalyzer
	analyseur d'accès : DeleteArchiveRule
	analyseur d'accès : GenerateFindingRecommendation
	analyseur d'accès : GetAccessPreview
	analyseur d'accès : GetAnalyzedResource
	analyseur d'accès : GetAnalyzer
	analyseur d'accès : GetArchiveRule

Préfixe de service	Actions
	analyseur d'accès : GetFinding
	analyseur d'accès : GetFindingRecommendation
	analyseur d'accès : GetFindingsStatistics
	analyseur d'accès : GetGeneratedPolicy
	analyseur d'accès : ListAccessPreviewFindings
	analyseur d'accès : ListAccessPreviews
	analyseur d'accès : ListAnalyzedResources
	analyseur d'accès : ListAnalyzers
	analyseur d'accès : ListArchiveRules
	analyseur d'accès : ListFindings
	analyseur d'accès : ListPolicyGenerations
	analyseur d'accès : StartPolicyGeneration
	analyseur d'accès : StartResourceScan
	analyseur d'accès : UpdateAnalyzer
	analyseur d'accès : UpdateArchiveRule
	analyseur d'accès : UpdateFindings
	analyseur d'accès : ValidatePolicy

Préfixe de service	Actions
compte	compte : AcceptPrimaryEmailUpdate compte : DeleteAlternateContact compte : DisableRegion compte : EnableRegion compte : GetAccountInformation compte : GetAlternateContact compte : GetContactInformation compte : GetGovCloudAccountInformation compte : GetPrimaryEmail compte : GetRegionOptStatus compte : ListRegions compte : PutAccountName compte : PutAlternateContact compte : PutContactInformation compte : StartPrimaryEmailUpdate

Préfixe de service	Actions
acm	ACM : CreateAcmeEndpoint
	ACM : DeleteCertificate
	ACM : DescribeCertificate
	ACM : ExportCertificate
	ACM : GetAccountConfiguration
	ACM : GetCertificate
	ACM : ImportCertificate
	ACM : ListAcmeEndpoints
	ACM : ListCertificates
	ACM : PutAccountConfiguration
	ACM : RenewCertificate
	ACM : RequestCertificate
	ACM : ResendValidationEmail
	ACM : SearchCertificates
	ACM : UpdateCertificateOptions

Préfixe de service	Actions
airflow	débit d'air : CreateCliToken débit d'air : CreateEnvironment débit d'air : CreateWebLoginToken débit d'air : DeleteEnvironment débit d'air : GetEnvironment débit d'air : ListEnvironments débit d'air : PublishMetrics débit d'air : UpdateEnvironment

Préfixe de service	Actions
amplify	amplifier : CreateApp
	amplifier : CreateBackendEnvironment
	amplifier : CreateBranch
	amplifier : CreateDeployment
	amplifier : CreateDomainAssociation
	amplifier : CreateWebHook
	amplifier : DeleteApp
	amplifier : DeleteBackendEnvironment
	amplifier : DeleteBranch
	amplifier : DeleteDomainAssociation
	amplifier : DeleteJob
	amplifier : DeleteWebHook
	amplifier : GenerateAccessLogs
	amplifier : GetApp
	amplifier : GetArtifactUrl
	amplifier : GetBackendEnvironment
	amplifier : GetBranch
	amplifier : GetDomainAssociation
	amplifier : GetJob
	amplifier : GetWebHook
	amplifier : ListApps

Préfixe de service	Actions
	amplifier : ListArtifacts amplifier : ListBackendEnvironments amplifier : ListBranches amplifier : ListDomainAssociations amplifier : ListJobs amplifier : ListWebHooks amplifier : StartDeployment amplifier : StartJob amplifier : StopJob amplifier : UpdateApp amplifier : UpdateBranch amplifier : UpdateDomainAssociation amplifier : UpdateWebHook

Préfixe de service	Actions
amplifyuibuilder	Amplify UI Builder : CreateComponent Amplify UI Builder : CreateForm Amplify UI Builder : CreateTheme Amplify UI Builder : DeleteComponent Amplify UI Builder : DeleteForm Amplify UI Builder : DeleteTheme Amplify UI Builder : ExportComponents Amplify UI Builder : ExportThemes Amplify UI Builder : GetCodegenJob Amplify UI Builder : ListCodegenJobs Amplify UI Builder : ListComponents Amplify UI Builder : ListForms Amplify UI Builder : ListThemes Amplify UI Builder : ResetMetadataFlag Amplify UI Builder : StartCodegenJob Amplify UI Builder : UpdateComponent Amplify UI Builder : UpdateForm Amplify UI Builder : UpdateTheme

Préfixe de service	Actions
app-integrations	intégrations d'applications : CreateApplication
	intégrations d'applications : CreateDataIntegration
	intégrations d'applications : CreateDataIntegrationAssociation
	intégrations d'applications : CreateEventIntegration
	intégrations d'applications : DeleteApplication
	intégrations d'applications : DeleteDataIntegration
	intégrations d'applications : DeleteEventIntegration
	intégrations d'applications : GetApplication
	intégrations d'applications : GetDataIntegration
	intégrations d'applications : GetEventIntegration
	intégrations d'applications : ListApplicationAssociations
	intégrations d'applications : ListApplications
	intégrations d'applications : ListDataIntegrationAssociations
	intégrations d'applications : ListDataIntegrations
	intégrations d'applications : ListEventIntegrationAssociations
	intégrations d'applications : ListEventIntegrations
	intégrations d'applications : UpdateApplication
	intégrations d'applications : UpdateDataIntegration
	intégrations d'applications : UpdateDataIntegrationAssociation
	intégrations d'applications : UpdateEventIntegration

Préfixe de service	Actions
appconfig	configuration de l'application : CreateApplication configuration de l'application : CreateConfigurationProfile configuration de l'application : CreateDeploymentStrategy configuration de l'application : CreateEnvironment configuration de l'application : CreateExtension configuration de l'application : CreateExtensionAssociation configuration de l'application : CreateHostedConfigurationVersion configuration de l'application : DeleteApplication configuration de l'application : DeleteConfigurationProfile configuration de l'application : DeleteDeploymentStrategy configuration de l'application : DeleteEnvironment configuration de l'application : DeleteExtension configuration de l'application : DeleteExtensionAssociation configuration de l'application : DeleteHostedConfigurationVersion configuration de l'application : GetAccountSettings configuration de l'application : GetApplication configuration de l'application : GetConfiguration configuration de l'application : GetConfigurationProfile configuration de l'application : GetDeployment configuration de l'application : GetDeploymentStrategy configuration de l'application : GetEnvironment

Préfixe de service	Actions
	configuration de l'application : GetExtension configuration de l'application : GetExtensionAssociation configuration de l'application : GetHostedConfigurationVersion configuration de l'application : ListApplications configuration de l'application : ListConfigurationProfiles configuration de l'application : ListDeploymentStrategies configuration de l'application : ListDeployments configuration de l'application : ListEnvironments configuration de l'application : ListExtensionAssociations configuration de l'application : ListExtensions configuration de l'application : ListHostedConfigurationVersions configuration de l'application : StartDeployment configuration de l'application : StopDeployment configuration de l'application : UpdateAccountSettings configuration de l'application : UpdateApplication configuration de l'application : UpdateConfigurationProfile configuration de l'application : UpdateDeploymentStrategy configuration de l'application : UpdateEnvironment configuration de l'application : UpdateExtension configuration de l'application : UpdateExtensionAssociation configuration de l'application : ValidateConfiguration

Préfixe de service	Actions
appflow	flux d'applications : CancelFlowExecutions
	flux d'applications : CreateConnectorProfile
	flux d'applications : CreateFlow
	flux d'applications : DeleteConnectorProfile
	flux d'applications : DeleteFlow
	flux d'applications : DescribeConnector
	flux d'applications : DescribeConnectorEntity
	flux d'applications : DescribeConnectorProfiles
	flux d'applications : DescribeConnectors
	flux d'applications : DescribeFlow
	flux d'applications : DescribeFlowExecutionRecords
	flux d'applications : ListConnectorEntities
	flux d'applications : ListConnectors
	flux d'applications : ListFlows
	flux d'applications : RegisterConnector
	flux d'applications : ResetConnectorMetadataCache
	flux d'applications : StartFlow
	flux d'applications : StopFlow
	flux d'applications : UnRegisterConnector
	flux d'applications : UpdateConnectorProfile
	flux d'applications : UpdateConnectorRegistration

Préfixe de service	Actions
	flux d'applications : UpdateFlow

Préfixe de service	Actions
applicationinsights	informations sur les applications : AddWorkload informations sur les applications : CreateApplication informations sur les applications : CreateComponent informations sur les applications : CreateLogPattern informations sur les applications : DeleteApplication informations sur les applications : DeleteComponent informations sur les applications : DeleteLogPattern informations sur les applications : DescribeApplication informations sur les applications : DescribeComponent informations sur les applications : DescribeComponentConfiguration informations sur les applications : DescribeComponentConfigurationRecommendation informations sur les applications : DescribeLogPattern informations sur les applications : DescribeObservation informations sur les applications : DescribeProblem informations sur les applications : DescribeProblemObservations informations sur les applications : DescribeWorkload informations sur les applications : ListApplications informations sur les applications : ListComponents informations sur les applications : ListConfigurationHistory informations sur les applications : ListLogPatternSets

Préfixe de service	Actions
	informations sur les applications : ListLogPatterns informations sur les applications : ListProblems informations sur les applications : ListWorkloads informations sur les applications : RemoveWorkload informations sur les applications : UpdateApplication informations sur les applications : UpdateComponent informations sur les applications : UpdateComponentConfiguration informations sur les applications : UpdateLogPattern informations sur les applications : UpdateWorkload

Préfixe de service	Actions
appmesh	maillage d'applications : CreateGatewayRoute
	maillage d'applications : CreateMesh
	maillage d'applications : CreateRoute
	maillage d'applications : CreateVirtualGateway
	maillage d'applications : CreateVirtualNode
	maillage d'applications : CreateVirtualRouter
	maillage d'applications : CreateVirtualService
	maillage d'applications : DeleteGatewayRoute
	maillage d'applications : DeleteMesh
	maillage d'applications : DeleteRoute
	maillage d'applications : DeleteVirtualGateway
	maillage d'applications : DeleteVirtualNode
	maillage d'applications : DeleteVirtualRouter
	maillage d'applications : DeleteVirtualService
	maillage d'applications : DescribeGatewayRoute
	maillage d'applications : DescribeMesh
	maillage d'applications : DescribeRoute
	maillage d'applications : DescribeVirtualGateway
	maillage d'applications : DescribeVirtualNode
	maillage d'applications : DescribeVirtualRouter
	maillage d'applications : DescribeVirtualService

Préfixe de service	Actions
	maillage d'applications : ListGatewayRoutes
	maillage d'applications : ListMeshes
	maillage d'applications : ListRoutes
	maillage d'applications : ListVirtualGateways
	maillage d'applications : ListVirtualNodes
	maillage d'applications : ListVirtualRouters
	maillage d'applications : ListVirtualServices
	maillage d'applications : StreamAggregatedResources
	maillage d'applications : UpdateGatewayRoute
	maillage d'applications : UpdateMesh
	maillage d'applications : UpdateRoute
	maillage d'applications : UpdateVirtualGateway
	maillage d'applications : UpdateVirtualNode
	maillage d'applications : UpdateVirtualRouter
	maillage d'applications : UpdateVirtualService

Préfixe de service	Actions
appstream	appstream : AssociateAppBlockBuilderAppBlock appstream : AssociateApplicationFleet appstream : AssociateApplicationToEntitlement appstream : AssociateFleet appstream : AssociateSoftwareToImageBuilder appstream : BatchAssociateUserStack appstream : BatchDisassociateUserStack appstream : CopyImage appstream : CreateAppBlock appstream : CreateAppBlockBuilder appstream : CreateAppBlockBuilderStreamingURL appstream : CreateApplication appstream : CreateDirectoryConfig appstream : CreateEntitlement appstream : CreateFleet appstream : CreateImageBuilder appstream : CreateImageBuilderStreamingURL appstream : CreateStack appstream : URL CreateStreaming appstream : CreateThemeForStack appstream : CreateUpdatedImage

Préfixe de service	Actions
	appstream : CreateUsageReportSubscription appstream : CreateUser appstream : DeleteAppBlock appstream : DeleteAppBlockBuilder appstream : DeleteApplication appstream : DeleteDirectoryConfig appstream : DeleteEntitlement appstream : DeleteFleet appstream : DeleteImage appstream : DeleteImageBuilder appstream : DeleteImagePermissions appstream : DeleteStack appstream : DeleteThemeForStack appstream : DeleteUsageReportSubscription appstream : DeleteUser appstream : DescribeAppBlockBuilderAppBlockAssociations appstream : DescribeAppBlockBuilders appstream : DescribeAppBlocks appstream : DescribeAppLicenseUsage appstream : DescribeApplicationFleetAssociations appstream : DescribeApplications

Préfixe de service	Actions
	appstream : DescribeDirectoryConfigs appstream : DescribeEntitlements appstream : DescribeFleets appstream : DescribeImageBuilders appstream : DescribeImagePermissions appstream : DescribeImages appstream : DescribeSessions appstream : DescribeStacks appstream : DescribeThemeForStack appstream : DescribeUsageReportSubscriptions appstream : DescribeUserStackAssociations appstream : DescribeUsers appstream : DisableUser appstream : DisassociateAppBlockBuilderAppBlock appstream : DisassociateApplicationFleet appstream : DisassociateApplicationFromEntitlement appstream : DisassociateFleet appstream : DisassociateSoftwareFromImageBuilder appstream : EnableUser appstream : ExpireSession appstream : GetExportImageTask

Préfixe de service	Actions
	appstream : ListAssociatedFleets appstream : ListAssociatedStacks appstream : ListEntitledApplications appstream : ListExportImageTasks appstream : StartAppBlockBuilder appstream : StartFleet appstream : StartImageBuilder appstream : StartSoftwareDeploymentToImageBuilder appstream : StopAppBlockBuilder appstream : StopFleet appstream : StopImageBuilder appstream : UpdateAppBlockBuilder appstream : UpdateApplication appstream : UpdateDirectoryConfig appstream : UpdateEntitlement appstream : UpdateFleet appstream : UpdateImagePermissions appstream : UpdateStack appstream : UpdateThemeForStack

Préfixe de service	Actions
appsync	synchronisation des applications : AssociateApi
	synchronisation des applications : AssociateMergedGraphqlApi
	synchronisation des applications : AssociateSourceGraphqlApi
	synchronisation des applications : CreateApi
	synchronisation des applications : CreateApiCache
	synchronisation des applications : CreateApiKey
	synchronisation des applications : CreateChannelNamespace
	synchronisation des applications : CreateDataSource
	synchronisation des applications : CreateDomainName
	synchronisation des applications : CreateFunction
	synchronisation des applications : CreateGraphqlApi
	synchronisation des applications : CreateResolver
	synchronisation des applications : CreateType
	synchronisation des applications : DeleteApi
	synchronisation des applications : DeleteApiCache
	synchronisation des applications : DeleteApiKey
	synchronisation des applications : DeleteChannelNamespace
	synchronisation des applications : DeleteDataSource
	synchronisation des applications : DeleteDomainName
	synchronisation des applications : DeleteFunction
	synchronisation des applications : DeleteGraphqlApi

Préfixe de service	Actions
	synchronisation des applications : DeleteResolver
	synchronisation des applications : DeleteType
	synchronisation des applications : DisassociateApi
	synchronisation des applications : DisassociateMergedGraphqlApi
	synchronisation des applications : DisassociateSourceGraphqlApi
	synchronisation des applications : EvaluateCode
	synchronisation des applications : EvaluateMappingTemplate
	synchronisation des applications : FlushApiCache
	synchronisation des applications : GetApi
	synchronisation des applications : GetApiAssociation
	synchronisation des applications : GetApiCache
	synchronisation des applications : GetChannelNamespace
	synchronisation des applications : GetDataSource
	synchronisation des applications : GetDataSourceIntrospection
	synchronisation des applications : GetDomainName
	synchronisation des applications : GetFunction
	synchronisation des applications : GetGraphqlApi
	synchronisation des applications : GetGraphqlApiEnvironmentVariables
	synchronisation des applications : GetIntrospectionSchema
	synchronisation des applications : GetResolver

Préfixe de service	Actions
	synchronisation des applications : GetSchemaCreationStatus
	synchronisation des applications : GetSourceApiAssociation
	synchronisation des applications : GetType
	synchronisation des applications : ListApiKeys
	synchronisation des applications : ListApis
	synchronisation des applications : ListChannelNamespaces
	synchronisation des applications : ListDataSources
	synchronisation des applications : ListDomainNames
	synchronisation des applications : ListFunctions
	synchronisation des applications : ListGraphqlApis
	synchronisation des applications : ListResolvers
	synchronisation des applications : ListResolversByFunction
	synchronisation des applications : ListSourceApiAssociations
	synchronisation des applications : ListTypes
	synchronisation des applications : ListTypesByAssociation
	synchronisation des applications : PutGraphqlApiEnvironmentVariables
	synchronisation des applications : StartDataSourceIntrospection
	synchronisation des applications : StartSchemaCreation
	synchronisation des applications : StartSchemaMerge
	synchronisation des applications : UpdateApi

Préfixe de service	Actions
	synchronisation des applications : UpdateApiCache
	synchronisation des applications : UpdateApiKey
	synchronisation des applications : UpdateChannelNamespace
	synchronisation des applications : UpdateDataSource
	synchronisation des applications : UpdateDomainName
	synchronisation des applications : UpdateFunction
	synchronisation des applications : UpdateGraphqlApi
	synchronisation des applications : UpdateResolver
	synchronisation des applications : UpdateSourceApiAssociation
	synchronisation des applications : UpdateType

Préfixe de service	Actions
aps	Cartes : CreateAlertManagerDefinition Cartes : CreateAnomalyDetector Cartes : CreateLoggingConfiguration Cartes : CreateQueryLoggingConfiguration Cartes : CreateRuleGroupsNamespace Cartes : CreateWorkspace Cartes : DeleteAlertManagerDefinition Cartes : DeleteAnomalyDetector Cartes : DeleteLoggingConfiguration Cartes : DeleteQueryLoggingConfiguration Cartes : DeleteResourcePolicy Cartes : DeleteRuleGroupsNamespace Cartes : DeleteScraper Cartes : DeleteScraperLoggingConfiguration Cartes : DeleteWorkspace Cartes : DescribeAlertManagerDefinition Cartes : DescribeAnomalyDetector Cartes : DescribeLoggingConfiguration Cartes : DescribeQueryLoggingConfiguration Cartes : DescribeResourcePolicy Cartes : DescribeRuleGroupsNamespace

Préfixe de service	Actions
	Cartes : DescribeScraper Cartes : DescribeScraperLoggingConfiguration Cartes : DescribeWorkspace Cartes : DescribeWorkspaceConfiguration Cartes : GetDefaultScraperConfiguration Cartes : ListAnomalyDetectors Cartes : ListRuleGroupsNamespaces Cartes : ListScrapers Cartes : ListWorkspaces Cartes : PutAlertManagerDefinition Cartes : PutAnomalyDetector Cartes : PutResourcePolicy Cartes : PutRuleGroupsNamespace Cartes : UpdateLoggingConfiguration Cartes : UpdateQueryLoggingConfiguration Cartes : UpdateScraper Cartes : UpdateScraperLoggingConfiguration Cartes : UpdateWorkspaceAlias Cartes : UpdateWorkspaceConfiguration

Préfixe de service	Actions
athéna	Athéna : BatchGetNamedQuery
	Athéna : BatchGetPreparedStatement
	Athéna : BatchGetQueryExecution
	Athéna : CancelCapacityReservation
	Athéna : CreateCapacityReservation
	Athéna : CreateDataCatalog
	Athéna : CreateNamedQuery
	Athéna : CreateNotebook
	Athéna : CreatePreparedStatement
	Athéna : CreatePresignedNotebookUrl
	Athéna : CreateWorkGroup
	Athéna : DeleteCapacityReservation
	Athéna : DeleteDataCatalog
	Athéna : DeleteNamedQuery
	Athéna : DeleteNotebook
	Athéna : DeletePreparedStatement
	Athéna : DeleteWorkGroup
	Athéna : ExportNotebook
	Athéna : GetCalculationExecution
	Athéna : GetCalculationExecutionCode
	Athéna : GetCalculationExecutionStatus

Préfixe de service	Actions
	Athéna : GetCapacityAssignmentConfiguration
	Athéna : GetCapacityReservation
	Athéna : GetDataCatalog
	Athéna : GetDatabase
	Athéna : GetNamedQuery
	Athéna : GetNotebookMetadata
	Athéna : GetPreparedStatement
	Athéna : GetQueryExecution
	Athéna : GetQueryResults
	Athéna : GetQueryResultsStream
	Athéna : GetQueryRuntimeStatistics
	Athéna : GetResourceDashboard
	Athéna : GetSession
	Athéna : GetSessionEndpoint
	Athéna : GetSessionStatus
	Athéna : GetTableMetadata
	Athéna : GetWorkGroup
	Athéna : ImportNotebook
	Athéna : ListApplicationDPUSizes
	Athéna : ListCalculationExecutions
	Athéna : ListCapacityReservations

Préfixe de service	Actions
	Athéna : ListDataCatalogs
	Athéna : ListDatabases
	Athéna : ListEngineVersions
	Athéna : ListExecutors
	Athéna : ListNamedQueries
	Athéna : ListNotebookMetadata
	Athéna : ListNotebookSessions
	Athéna : ListPreparedStatements
	Athéna : ListQueryExecutions
	Athéna : ListSessions
	Athéna : ListTableMetadata
	Athéna : ListWorkGroups
	Athéna : PutCapacityAssignmentConfiguration
	Athéna : StartCalculationExecution
	Athéna : StartQueryExecution
	Athéna : StartSession
	Athéna : StopCalculationExecution
	Athéna : StopQueryExecution
	Athéna : TerminateSession
	Athéna : UpdateCapacityReservation
	Athéna : UpdateDataCatalog

Préfixe de service	Actions
	Athéna : UpdateNamedQuery Athéna : UpdateNotebook Athéna : UpdateNotebookMetadata Athéna : UpdatePreparedStatement Athéna : UpdateWorkGroup

Préfixe de service	Actions
auditmanager	responsable de l'audit : AssociateAssessmentReportEvidenceFolder responsable de l'audit : BatchAssociateAssessmentReportEvidence responsable de l'audit : BatchCreateDelegationByAssessment responsable de l'audit : BatchDeleteDelegationByAssessment responsable de l'audit : BatchDisassociateAssessmentReportEvidence responsable de l'audit : BatchImportEvidenceToAssessmentControl responsable de l'audit : CreateAssessment responsable de l'audit : CreateAssessmentFramework responsable de l'audit : CreateAssessmentReport responsable de l'audit : CreateControl responsable de l'audit : DeleteAssessment responsable de l'audit : DeleteAssessmentFramework responsable de l'audit : DeleteAssessmentFrameworkShare responsable de l'audit : DeleteAssessmentReport responsable de l'audit : DeleteControl responsable de l'audit : DeregisterAccount responsable de l'audit : DeregisterOrganizationAdminAccount responsable de l'audit : DisassociateAssessmentReportEvidenceFolder responsable de l'audit : GetAccountStatus

Préfixe de service	Actions
	responsable de l'audit : GetAssessment responsable de l'audit : GetAssessmentFramework responsable de l'audit : GetAssessmentReportUrl responsable de l'audit : GetChangeLogs responsable de l'audit : GetControl responsable de l'audit : GetDelegations responsable de l'audit : GetEvidence responsable de l'audit : GetEvidenceByEvidenceFolder responsable de l'audit : GetEvidenceFileUploadUrl responsable de l'audit : GetEvidenceFolder responsable de l'audit : GetEvidenceFoldersByAssessment responsable de l'audit : GetEvidenceFoldersByAssessmentControl responsable de l'audit : GetInsights responsable de l'audit : GetInsightsByAssessment responsable de l'audit : GetOrganizationAdminAccount responsable de l'audit : GetServicesInScope responsable de l'audit : GetSettings responsable de l'audit : ListAssessmentControlInsightsByControlDomain responsable de l'audit : ListAssessmentFrameworkShareRequests responsable de l'audit : ListAssessmentFrameworks

Préfixe de service	Actions
	responsable de l'audit : ListAssessmentReports responsable de l'audit : ListAssessments responsable de l'audit : ListControlDomainInsights responsable de l'audit : ListControlDomainInsightsByAssessment responsable de l'audit : ListControlInsightsByControlDomain responsable de l'audit : ListControls responsable de l'audit : ListKeywordsForDataSource responsable de l'audit : ListNotifications responsable de l'audit : RegisterAccount responsable de l'audit : RegisterOrganizationAdminAccount responsable de l'audit : StartAssessmentFrameworkShare responsable de l'audit : UpdateAssessment responsable de l'audit : UpdateAssessmentControl responsable de l'audit : UpdateAssessmentControlSetStatus responsable de l'audit : UpdateAssessmentFramework responsable de l'audit : UpdateAssessmentFrameworkShare responsable de l'audit : UpdateAssessmentStatus responsable de l'audit : UpdateControl responsable de l'audit : UpdateSettings responsable de l'audit : ValidateAssessmentReportIntegrity

Préfixe de service	Actions
autoscaling	mise à l'échelle automatique : AttachInstances
	mise à l'échelle automatique : AttachLoadBalancerTargetGroups
	mise à l'échelle automatique : AttachLoadBalancers
	mise à l'échelle automatique : AttachTrafficSources
	mise à l'échelle automatique : BatchDeleteScheduledAction
	mise à l'échelle automatique : BatchPutScheduledUpdateGroupAction
	mise à l'échelle automatique : CancelInstanceRefresh
	mise à l'échelle automatique : CompleteLifecycleAction
	mise à l'échelle automatique : CreateAutoScalingGroup
	mise à l'échelle automatique : CreateLaunchConfiguration
	mise à l'échelle automatique : DeleteAutoScalingGroup
	mise à l'échelle automatique : DeleteLaunchConfiguration
	mise à l'échelle automatique : DeleteLifecycleHook
	mise à l'échelle automatique : DeleteNotificationConfiguration
	mise à l'échelle automatique : DeletePolicy
	mise à l'échelle automatique : DeleteScheduledAction
	mise à l'échelle automatique : DeleteWarmPool
	mise à l'échelle automatique : DescribeAccountLimits
	mise à l'échelle automatique : DescribeAdjustmentTypes
	mise à l'échelle automatique : DescribeAutoScalingGroups

Préfixe de service	Actions
	mise à l'échelle automatique : DescribeAutoScalingInstances
	mise à l'échelle automatique : DescribeAutoScalingNotificationTypes
	mise à l'échelle automatique : DescribeInstanceRefreshes
	mise à l'échelle automatique : DescribeLaunchConfigurations
	mise à l'échelle automatique : DescribeLifecycleHookTypes
	mise à l'échelle automatique : DescribeLifecycleHooks
	mise à l'échelle automatique : DescribeLoadBalancerTargetGroups
	mise à l'échelle automatique : DescribeLoadBalancers
	mise à l'échelle automatique : DescribeMetricCollectionTypes
	mise à l'échelle automatique : DescribeNotificationConfigurations
	mise à l'échelle automatique : DescribePolicies
	mise à l'échelle automatique : DescribeScalingActivities
	mise à l'échelle automatique : DescribeScalingProcessTypes
	mise à l'échelle automatique : DescribeScheduledActions
	mise à l'échelle automatique : DescribeTerminationPolicyTypes
	mise à l'échelle automatique : DescribeTrafficSources
	mise à l'échelle automatique : DescribeWarmPool
	mise à l'échelle automatique : DetachInstances
	mise à l'échelle automatique : DetachLoadBalancerTargetGroups
	mise à l'échelle automatique : DetachLoadBalancers

Préfixe de service	Actions
	mise à l'échelle automatique : DetachTrafficSources
	mise à l'échelle automatique : DisableMetricsCollection
	mise à l'échelle automatique : EnableMetricsCollection
	mise à l'échelle automatique : EnterStandby
	mise à l'échelle automatique : ExecutePolicy
	mise à l'échelle automatique : ExitStandby
	mise à l'échelle automatique : GetPredictiveScalingForecast
	mise à l'échelle automatique : PutLifecycleHook
	mise à l'échelle automatique : PutNotificationConfiguration
	mise à l'échelle automatique : PutScalingPolicy
	mise à l'échelle automatique : PutScheduledUpdateGroupAction
	mise à l'échelle automatique : PutWarmPool
	mise à l'échelle automatique : RecordLifecycleActionHeartbeat
	mise à l'échelle automatique : ResumeProcesses
	mise à l'échelle automatique : RollbackInstanceRefresh
	mise à l'échelle automatique : SetDesiredCapacity
	mise à l'échelle automatique : SetInstanceHealth
	mise à l'échelle automatique : SetInstanceProtection
	mise à l'échelle automatique : StartInstanceRefresh
	mise à l'échelle automatique : SuspendProcesses

Préfixe de service	Actions
	mise à l'échelle automatique : TerminateInstanceInAutoScalingGroup mise à l'échelle automatique : UpdateAutoScalingGroup
aws-marketplace	AWS Marketplace : GetEntitlements

Préfixe de service	Actions
sauvegarde	sauvegarde : CancelLegalHold
	sauvegarde : CreateBackupPlan
	sauvegarde : CreateBackupSelection
	sauvegarde : CreateBackupVault
	sauvegarde : CreateFramework
	sauvegarde : CreateLegalHold
	sauvegarde : CreateLogicallyAirGappedBackupVault
	sauvegarde : CreateReportPlan
	sauvegarde : CreateRestoreAccessBackupVault
	sauvegarde : CreateRestoreTestingPlan
	sauvegarde : CreateRestoreTestingSelection
	sauvegarde : CreateTieringConfiguration
	sauvegarde : DeleteBackupPlan
	sauvegarde : DeleteBackupSelection
	sauvegarde : DeleteBackupVault
	sauvegarde : DeleteBackupVaultAccessPolicy
	sauvegarde : DeleteBackupVaultLockConfiguration
	sauvegarde : DeleteBackupVaultNotifications
	sauvegarde : DeleteFramework
	sauvegarde : DeleteRecoveryPoint
	sauvegarde : DeleteReportPlan

Préfixe de service	Actions
	sauvegarde : DeleteRestoreTestingPlan
	sauvegarde : DeleteRestoreTestingSelection
	sauvegarde : DeleteTieringConfiguration
	sauvegarde : DescribeBackupJob
	sauvegarde : DescribeBackupVault
	sauvegarde : DescribeCopyJob
	sauvegarde : DescribeFramework
	sauvegarde : DescribeGlobalSettings
	sauvegarde : DescribeProtectedResource
	sauvegarde : DescribeRecoveryPoint
	sauvegarde : DescribeRegionSettings
	sauvegarde : DescribeReportJob
	sauvegarde : DescribeReportPlan
	sauvegarde : DescribeRestoreJob
	sauvegarde : DescribeScanJob
	sauvegarde : DisassociateRecoveryPoint
	sauvegarde : DisassociateRecoveryPointFromParent
	sauvegarde : ExportBackupPlanTemplate
	sauvegarde : GetBackupPlan
	sauvegarde : GetBackupPlanFromJSON
	sauvegarde : GetBackupPlanFromTemplate

Préfixe de service	Actions
	sauvegarde : GetBackupSelection
	sauvegarde : GetBackupVaultAccessPolicy
	sauvegarde : GetBackupVaultNotifications
	sauvegarde : GetLegalHold
	sauvegarde : GetRecoveryPointRestoreMetadata
	sauvegarde : GetRestoreJobMetadata
	sauvegarde : GetRestoreTestingInferredMetadata
	sauvegarde : GetRestoreTestingPlan
	sauvegarde : GetRestoreTestingSelection
	sauvegarde : GetSupportedResourceTypes
	sauvegarde : GetTieringConfiguration
	sauvegarde : ListBackupJobSummaries
	sauvegarde : ListBackupJobs
	sauvegarde : ListBackupPlanTemplates
	sauvegarde : ListBackupPlanVersions
	sauvegarde : ListBackupPlans
	sauvegarde : ListBackupSelections
	sauvegarde : ListBackupVaults
	sauvegarde : ListCopyJobSummaries
	sauvegarde : ListCopyJobs
	sauvegarde : ListFrameworks

Préfixe de service	Actions
	sauvegarde : ListIndexedRecoveryPoints
	sauvegarde : ListLegalHolds
	sauvegarde : ListProtectedResources
	sauvegarde : ListRecoveryPointsByBackupVault
	sauvegarde : ListRecoveryPointsByLegalHold
	sauvegarde : ListRecoveryPointsByResource
	sauvegarde : ListReportJobs
	sauvegarde : ListReportPlans
	sauvegarde : ListRestoreAccessBackupVaults
	sauvegarde : ListRestoreJobSummaries
	sauvegarde : ListRestoreJobs
	sauvegarde : ListRestoreJobsByProtectedResource
	sauvegarde : ListRestoreTestingPlans
	sauvegarde : ListRestoreTestingSelections
	sauvegarde : ListScanJobSummaries
	sauvegarde : ListScanJobs
	sauvegarde : ListTieringConfigurations
	sauvegarde : PutBackupVaultAccessPolicy
	sauvegarde : PutBackupVaultLockConfiguration
	sauvegarde : PutBackupVaultNotifications
	sauvegarde : PutRestoreValidationResult

Préfixe de service	Actions
	sauvegarde : StartBackupJob
	sauvegarde : StartCopyJob
	sauvegarde : StartReportJob
	sauvegarde : StartRestoreJob
	sauvegarde : StopBackupJob
	sauvegarde : UpdateBackupPlan
	sauvegarde : UpdateFramework
	sauvegarde : UpdateGlobalSettings
	sauvegarde : UpdateRecoveryPointLifecycle
	sauvegarde : UpdateRegionSettings
	sauvegarde : UpdateReportPlan
	sauvegarde : UpdateRestoreTestingPlan
	sauvegarde : UpdateRestoreTestingSelection
	sauvegarde : UpdateTieringConfiguration

Préfixe de service	Actions
lot	lot : CancelJob lot : CreateComputeEnvironment lot : CreateConsumableResource lot : CreateJobQueue lot : CreateQuotaShare lot : CreateSchedulingPolicy lot : CreateServiceEnvironment lot : DeleteComputeEnvironment lot : DeleteConsumableResource lot : DeleteJobQueue lot : DeleteQuotaShare lot : DeleteSchedulingPolicy lot : DeleteServiceEnvironment lot : DeregisterJobDefinition lot : DescribeComputeEnvironments lot : DescribeConsumableResource lot : DescribeJobDefinitions lot : DescribeJobQueues lot : DescribeJobs lot : DescribeQuotaShare lot : DescribeSchedulingPolicies

Préfixe de service	Actions
	lot : DescribeServiceEnvironments lot : DescribeServiceJob lot : GetJobQueueSnapshot lot : ListConsumableResources lot : ListJobs lot : ListJobsByConsumableResource lot : ListQuotaShares lot : ListSchedulingPolicies lot : ListServiceJobs lot : RegisterJobDefinition lot : SubmitJob lot : SubmitServiceJob lot : TerminateJob lot : TerminateServiceJob lot : UpdateComputeEnvironment lot : UpdateConsumableResource lot : UpdateJobQueue lot : UpdateQuotaShare lot : UpdateSchedulingPolicy lot : UpdateServiceEnvironment lot : UpdateServiceJob

Préfixe de service	Actions
braket	support : CancelJob support : CancelQuantumTask support : CreateJob support : CreateQuantumTask support : CreateSpendingLimit support : GetDevice support : GetJob support : GetQuantumTask support : SearchDevices support : SearchJobs support : SearchQuantumTasks support : SearchSpendingLimits

Préfixe de service	Actions
Budgets	budgets : CreateBudgetAction budgets : DeleteBudgetAction budgets : DescribeBudgetAction budgets : DescribeBudgetActionHistories budgets : DescribeBudgetActionsForAccount budgets : DescribeBudgetActionsForBudget budgets : ExecuteBudgetAction budgets : ModifyBudget budgets : UpdateBudgetAction budgets : ViewBudget
Cloud9	cloud 9 : EC2 CreateEnvironment nuage 9 : CreateEnvironmentMembership nuage 9 : DeleteEnvironment nuage 9 : DeleteEnvironmentMembership nuage 9 : DescribeEnvironmentMemberships nuage 9 : DescribeEnvironmentStatus nuage 9 : DescribeEnvironments nuage 9 : ListEnvironments nuage 9 : UpdateEnvironment nuage 9 : UpdateEnvironmentMembership

Préfixe de service	Actions
cloudformation	formation des nuages : BatchDescribeTypeConfigurations
	formation des nuages : CancelUpdateStack
	formation des nuages : ContinueUpdateRollback
	formation des nuages : CreateChangeSet
	formation des nuages : CreateGeneratedTemplate
	formation des nuages : CreateStack
	formation des nuages : CreateStackInstances
	formation des nuages : CreateStackSet
	formation des nuages : DeactivateType
	formation des nuages : DeleteChangeSet
	formation des nuages : DeleteGeneratedTemplate
	formation des nuages : DeleteStack
	formation des nuages : DeleteStackInstances
	formation des nuages : DeleteStackSet
	formation des nuages : DeregisterType
	formation des nuages : DescribeAccountLimits
	formation des nuages : DescribeChangeSet
	formation des nuages : DescribeChangeSetHooks
	formation des nuages : DescribeEvents
	formation des nuages : DescribeGeneratedTemplate
	formation des nuages : DescribeOrganizationsAccess

Préfixe de service	Actions
	formation des nuages : DescribePublisher
	formation des nuages : DescribeResourceScan
	formation des nuages : DescribeStackDriftDetectionStatus
	formation des nuages : DescribeStackEvents
	formation des nuages : DescribeStackInstance
	formation des nuages : DescribeStackResource
	formation des nuages : DescribeStackResourceDrifts
	formation des nuages : DescribeStackResources
	formation des nuages : DescribeStackSet
	formation des nuages : DescribeStackSetOperation
	formation des nuages : DescribeStacks
	formation des nuages : DescribeType
	formation des nuages : DescribeTypeRegistration
	formation des nuages : DetectStackDrift
	formation des nuages : DetectStackResourceDrift
	formation des nuages : DetectStackSetDrift
	formation des nuages : EstimateTemplateCost
	formation des nuages : ExecuteChangeSet
	formation des nuages : GetGeneratedTemplate
	formation des nuages : GetHookResult
	formation des nuages : GetStackPolicy

Préfixe de service	Actions
	formation des nuages : GetTemplate
	formation des nuages : GetTemplateSummary
	formation des nuages : ImportStacksToStackSet
	formation des nuages : ListChangeSets
	formation des nuages : ListExports
	formation des nuages : ListGeneratedTemplates
	formation des nuages : ListHookResults
	formation des nuages : ListImports
	formation des nuages : ListResourceScanRelatedResources
	formation des nuages : ListResourceScanResources
	formation des nuages : ListResourceScans
	formation des nuages : ListStackInstanceResourceDrifts
	formation des nuages : ListStackInstances
	formation des nuages : ListStackRefactors
	formation des nuages : ListStackResources
	formation des nuages : ListStackSetAutoDeploymentTargets
	formation des nuages : ListStackSetOperationResults
	formation des nuages : ListStackSetOperations
	formation des nuages : ListStackSets
	formation des nuages : ListTypeRegistrations
	formation des nuages : ListTypeVersions

Préfixe de service	Actions
	formation des nuages : ListTypes formation des nuages : PublishType formation des nuages : RecordHandlerProgress formation des nuages : RegisterPublisher formation des nuages : RegisterType formation des nuages : RollbackStack formation des nuages : SetStackPolicy formation des nuages : SetTypeConfiguration formation des nuages : SetTypeDefaultVersion formation des nuages : SignalResource formation des nuages : StartResourceScan formation des nuages : StopStackSetOperation formation des nuages : TestType formation des nuages : UpdateGeneratedTemplate formation des nuages : UpdateStack formation des nuages : UpdateStackInstances formation des nuages : UpdateStackSet formation des nuages : UpdateTerminationProtection formation des nuages : ValidateTemplate

Préfixe de service	Actions
cloudfront	front de nuage : AssociateAlias
	front de nuage : AssociateDistributionTenantWebACL
	front de nuage : AssociateDistributionWebACL
	front de nuage : CreateCachePolicy
	front de nuage : CreateCloudFrontOriginAccessIdentity
	front de nuage : CreateConnectionFunction
	front de nuage : CreateContinuousDeploymentPolicy
	front de nuage : CreateDistributionTenant
	front de nuage : CreateFieldLevelEncryptionConfig
	front de nuage : CreateFieldLevelEncryptionProfile
	front de nuage : CreateFunction
	front de nuage : CreateInvalidation
	front de nuage : CreateKeyGroup
	front de nuage : CreateKeyValueStore
	front de nuage : CreateMonitoringSubscription
	front de nuage : CreateOriginAccessControl
	front de nuage : CreateOriginRequestPolicy
	front de nuage : CreatePublicKey
	front de nuage : CreateRealtimeLogConfig
	front de nuage : CreateResponseHeadersPolicy
	front de nuage : CreateTrustStore

Préfixe de service	Actions
	front de nuage : DeleteAnycastIpList
	front de nuage : DeleteCachePolicy
	front de nuage : DeleteCloudFrontOriginAccessIdentity
	front de nuage : DeleteConnectionFunction
	front de nuage : DeleteConnectionGroup
	front de nuage : DeleteContinuousDeploymentPolicy
	front de nuage : DeleteDistribution
	front de nuage : DeleteDistributionTenant
	front de nuage : DeleteFieldLevelEncryptionConfig
	front de nuage : DeleteFieldLevelEncryptionProfile
	front de nuage : DeleteFunction
	front de nuage : DeleteKeyGroup
	front de nuage : DeleteKeyValueStore
	front de nuage : DeleteMonitoringSubscription
	front de nuage : DeleteOriginAccessControl
	front de nuage : DeleteOriginRequestPolicy
	front de nuage : DeletePublicKey
	front de nuage : DeleteRealtimeLogConfig
	front de nuage : DeleteResponseHeadersPolicy
	front de nuage : DeleteStreamingDistribution
	front de nuage : DeleteTrustStore

Préfixe de service	Actions
	front de nuage : DeleteVpcOrigin
	front de nuage : DescribeFunction
	front de nuage : DescribeKeyValueStore
	front de nuage : DisassociateDistributionTenantWebACL
	front de nuage : DisassociateDistributionWebACL
	front de nuage : GetAnycastIpList
	front de nuage : GetCachePolicy
	front de nuage : GetCachePolicyConfig
	front de nuage : GetCloudFrontOriginAccessIdentity
	front de nuage : GetCloudFrontOriginAccessIdentityConfig
	front de nuage : GetContinuousDeploymentPolicy
	front de nuage : GetContinuousDeploymentPolicyConfig
	front de nuage : GetDistributionConfig
	front de nuage : GetFieldLevelEncryption
	front de nuage : GetFieldLevelEncryptionConfig
	front de nuage : GetFieldLevelEncryptionProfile
	front de nuage : GetFieldLevelEncryptionProfileConfig
	front de nuage : GetFunction
	front de nuage : GetInvalidation
	front de nuage : GetInvalidationForDistributionTenant
	front de nuage : GetKeyGroup

Préfixe de service	Actions
	front de nuage : GetKeyGroupConfig
	front de nuage : GetMonitoringSubscription
	front de nuage : GetOriginAccessControl
	front de nuage : GetOriginAccessControlConfig
	front de nuage : GetOriginRequestPolicy
	front de nuage : GetOriginRequestPolicyConfig
	front de nuage : GetPublicKey
	front de nuage : GetPublicKeyConfig
	front de nuage : GetRealtimeLogConfig
	front de nuage : GetResponseHeadersPolicy
	front de nuage : GetResponseHeadersPolicyConfig
	front de nuage : GetStreamingDistribution
	front de nuage : GetStreamingDistributionConfig
	front de nuage : GetVpcOrigin
	front de nuage : ListAnycastIpLists
	front de nuage : ListCachePolicies
	front de nuage : ListCloudFrontOriginAccessIdentities
	front de nuage : ListConflictingAliases
	front de nuage : ListConnectionFunctions
	front de nuage : ListConnectionGroups
	front de nuage : ListContinuousDeploymentPolicies

Préfixe de service	Actions
	front de nuage : ListDistributionTenants
	front de nuage : ListDistributionTenantsByCustomization
	front de nuage : ListDistributions
	front de nuage : ListDistributionsByAnycastIpListId
	front de nuage : ListDistributionsByCachePolicyId
	front de nuage : ListDistributionsByConnectionMode
	front de nuage : ListDistributionsByKeyGroup
	front de nuage : ListDistributionsByOriginRequestPolicyId
	front de nuage : ListDistributionsByRealtimeLogConfig
	front de nuage : ListDistributionsByResponseHeadersPolicyId
	front de nuage : ListDistributionsByVpcOriginId
	front de nuage : ListDistributionsByWebACLId
	front de nuage : ListFieldLevelEncryptionConfigs
	front de nuage : ListFieldLevelEncryptionProfiles
	front de nuage : ListFunctions
	front de nuage : ListInvalidations
	front de nuage : ListInvalidationsForDistributionTenant
	front de nuage : ListKeyGroups
	front de nuage : ListKeyValueStores
	front de nuage : ListOriginAccessControls
	front de nuage : ListOriginRequestPolicies

Préfixe de service	Actions
	front de nuage : ListPublicKeys
	front de nuage : ListRealtimeLogConfigs
	front de nuage : ListResponseHeadersPolicies
	front de nuage : ListStreamingDistributions
	front de nuage : ListTrustStores
	front de nuage : PublishConnectionFunction
	front de nuage : PublishFunction
	front de nuage : TestConnectionFunction
	front de nuage : TestFunction
	front de nuage : UpdateAnycastIpList
	front de nuage : UpdateCachePolicy
	front de nuage : UpdateCloudFrontOriginAccessIdentity
	front de nuage : UpdateConnectionFunction
	front de nuage : UpdateConnectionGroup
	front de nuage : UpdateContinuousDeploymentPolicy
	front de nuage : UpdateDistribution
	front de nuage : UpdateDistributionTenant
	front de nuage : UpdateFieldLevelEncryptionConfig
	front de nuage : UpdateFieldLevelEncryptionProfile
	front de nuage : UpdateFunction
	front de nuage : UpdateKeyGroup

Préfixe de service	Actions
	front de nuage : UpdateKeyValueStore front de nuage : UpdateOriginAccessControl front de nuage : UpdateOriginRequestPolicy front de nuage : UpdatePublicKey front de nuage : UpdateRealtimeLogConfig front de nuage : UpdateResponseHeadersPolicy front de nuage : UpdateTrustStore
cloudhsm	cloudhsm : CreateHsm cloudhsm : DeleteBackup cloudhsm : DeleteHsm cloudhsm : DeleteResourcePolicy cloudhsm : DescribeBackups cloudhsm : DescribeClusters cloudhsm : GetResourcePolicy cloudhsm : InitializeCluster cloudhsm : ModifyBackupAttributes cloudhsm : ModifyCluster cloudhsm : PutResourcePolicy cloudhsm : RestoreBackup

Préfixe de service	Actions
cloudsearch	recherche dans le cloud : BuildSuggesters
	recherche dans le cloud : CreateDomain
	recherche dans le cloud : DefineAnalysisScheme
	recherche dans le cloud : DefineExpression
	recherche dans le cloud : DefineIndexField
	recherche dans le cloud : DefineSuggester
	recherche dans le cloud : DeleteAnalysisScheme
	recherche dans le cloud : DeleteDomain
	recherche dans le cloud : DeleteExpression
	recherche dans le cloud : DeleteIndexField
	recherche dans le cloud : DeleteSuggester
	recherche dans le cloud : DescribeAnalysisSchemes
	recherche dans le cloud : DescribeAvailabilityOptions
	recherche dans le cloud : DescribeDomainEndpointOptions
	recherche dans le cloud : DescribeDomains
	recherche dans le cloud : DescribeExpressions
	recherche dans le cloud : DescribeIndexFields
	recherche dans le cloud : DescribeScalingParameters
	recherche dans le cloud : DescribeServiceAccessPolicies
	recherche dans le cloud : DescribeSuggesters
	recherche dans le cloud : IndexDocuments

Préfixe de service	Actions
	recherche dans le cloud : ListDomainNames
	recherche dans le cloud : UpdateAvailabilityOptions
	recherche dans le cloud : UpdateDomainEndpointOptions
	recherche dans le cloud : UpdateScalingParameters
	recherche dans le cloud : UpdateServiceAccessPolicies

Préfixe de service	Actions
cloudtrail	traînée nuageuse : CancelQuery traînée nuageuse : CreateChannel traînée nuageuse : CreateDashboard traînée nuageuse : CreateEventDataStore traînée nuageuse : CreateTrail traînée nuageuse : DeleteChannel traînée nuageuse : DeleteDashboard traînée nuageuse : DeleteEventDataStore traînée nuageuse : DeleteResourcePolicy traînée nuageuse : DeleteTrail traînée nuageuse : DeregisterOrganizationDelegatedAdmin traînée nuageuse : DescribeQuery traînée nuageuse : DescribeTrails traînée nuageuse : DisableFederation traînée nuageuse : GenerateQuery traînée nuageuse : GetChannel traînée nuageuse : GetDashboard traînée nuageuse : GetEventConfiguration traînée nuageuse : GetEventDataStore traînée nuageuse : GetEventDataStoreData traînée nuageuse : GetEventSelectors

Préfixe de service	Actions
	traînée nuageuse : GetImport traînée nuageuse : GetInsightSelectors traînée nuageuse : GetResourcePolicy traînée nuageuse : GetTrail traînée nuageuse : GetTrailStatus traînée nuageuse : ListChannels traînée nuageuse : ListDashboards traînée nuageuse : ListEventDataStores traînée nuageuse : ListImportFailures traînée nuageuse : ListImports traînée nuageuse : ListInsightsData traînée nuageuse : ListPublicKeys traînée nuageuse : ListQueries traînée nuageuse : ListTrails traînée nuageuse : LookupEvents traînée nuageuse : PutEventConfiguration traînée nuageuse : PutEventSelectors traînée nuageuse : PutInsightSelectors traînée nuageuse : PutResourcePolicy traînée nuageuse : RegisterOrganizationDelegatedAdmin traînée nuageuse : RestoreEventDataStore

Préfixe de service	Actions
	traînée nuageuse : SearchSampleQueries traînée nuageuse : StartEventDataStoreIngestion traînée nuageuse : StartImport traînée nuageuse : StartLogging traînée nuageuse : StartQuery traînée nuageuse : StopEventDataStoreIngestion traînée nuageuse : StopImport traînée nuageuse : StopLogging traînée nuageuse : UpdateChannel traînée nuageuse : UpdateDashboard traînée nuageuse : UpdateEventDataStore traînée nuageuse : UpdateTrail

Préfixe de service	Actions
cloudwatch	surveillance des nuages : DeleteAlarmMuteRule
	surveillance des nuages : DeleteAlarms
	surveillance des nuages : DeleteAnomalyDetector
	surveillance des nuages : DeleteDashboards
	surveillance des nuages : DeleteInsightRules
	surveillance des nuages : DeleteMetricStream
	surveillance des nuages : DescribeAlarmHistory
	surveillance des nuages : DescribeAlarms
	surveillance des nuages : DescribeAlarmsForMetric
	surveillance des nuages : DescribeAnomalyDetectors
	surveillance des nuages : DescribeInsightRules
	surveillance des nuages : DisableAlarmActions
	surveillance des nuages : DisableInsightRules
	surveillance des nuages : EnableAlarmActions
	surveillance des nuages : EnableInsightRules
	surveillance des nuages : GetAlarmMuteRule
	surveillance des nuages : GetDashboard
	surveillance des nuages : GetDataset
	surveillance des nuages : GetInsightRuleReport
	surveillance des nuages : GetMetricStatistics
	surveillance des nuages : GetMetricStream

Préfixe de service	Actions
	surveillance des nuages : ListAlarmMuteRules
	surveillance des nuages : ListDashboards
	surveillance des nuages : ListManagedInsightRules
	surveillance des nuages : ListMetricStreams
	surveillance des nuages : PutAlarmMuteRule
	surveillance des nuages : PutAnomalyDetector
	surveillance des nuages : PutCompositeAlarm
	surveillance des nuages : PutDashboard
	surveillance des nuages : PutInsightRule
	surveillance des nuages : PutLogAlarm
	surveillance des nuages : PutManagedInsightRules
	surveillance des nuages : PutMetricAlarm
	surveillance des nuages : PutMetricStream
	surveillance des nuages : SetAlarmState
	surveillance des nuages : StartMetricStreams
	surveillance des nuages : StopMetricStreams

Préfixe de service	Actions
codeartifact	artefact de code : AssociateExternalConnection artefact de code : CopyPackageVersions artefact de code : CreateDomain artefact de code : CreateRepository artefact de code : DeleteDomain artefact de code : DeleteDomainPermissionsPolicy artefact de code : DeletePackage artefact de code : DeletePackageVersions artefact de code : DeleteRepository artefact de code : DeleteRepositoryPermissionsPolicy artefact de code : DescribeDomain artefact de code : DescribePackage artefact de code : DescribePackageVersion artefact de code : DescribeRepository artefact de code : DisassociateExternalConnection artefact de code : DisposePackageVersions artefact de code : GetAssociatedPackageGroup artefact de code : GetAuthorizationToken artefact de code : GetDomainPermissionsPolicy artefact de code : GetPackageVersionAsset artefact de code : GetPackageVersionReadme

Préfixe de service	Actions
	artefact de code : GetRepositoryEndpoint artefact de code : GetRepositoryPermissionsPolicy artefact de code : ListDomains artefact de code : ListPackageGroups artefact de code : ListPackageVersionAssets artefact de code : ListPackageVersionDependencies artefact de code : ListPackageVersions artefact de code : ListPackages artefact de code : ListRepositories artefact de code : ListRepositoriesInDomain artefact de code : PublishPackageVersion artefact de code : PutDomainPermissionsPolicy artefact de code : PutPackageMetadata artefact de code : PutPackageOriginConfiguration artefact de code : PutRepositoryPermissionsPolicy artefact de code : ReadFromRepository artefact de code : UpdatePackageVersionsStatus artefact de code : UpdateRepository

Préfixe de service	Actions
codedeploy	déploiement de code : BatchGetApplicationRevisions
	déploiement de code : BatchGetApplications
	déploiement de code : BatchGetDeploymentGroups
	déploiement de code : BatchGetDeploymentInstances
	déploiement de code : BatchGetDeploymentTargets
	déploiement de code : BatchGetDeployments
	déploiement de code : BatchGetOnPremisesInstances
	déploiement de code : ContinueDeployment
	déploiement de code : CreateApplication
	déploiement de code : CreateDeployment
	déploiement de code : CreateDeploymentConfig
	déploiement de code : CreateDeploymentGroup
	déploiement de code : DeleteApplication
	déploiement de code : DeleteDeploymentConfig
	déploiement de code : DeleteDeploymentGroup
	déploiement de code : DeleteGitHubAccountToken
	déploiement de code : DeleteResourcesByExternalId
	déploiement de code : DeregisterOnPremisesInstance
	déploiement de code : GetApplication
	déploiement de code : GetApplicationRevision
	déploiement de code : GetDeployment

Préfixe de service	Actions
	déploiement de code : GetDeploymentConfig
	déploiement de code : GetDeploymentGroup
	déploiement de code : GetDeploymentInstance
	déploiement de code : GetDeploymentTarget
	déploiement de code : GetOnPremisesInstance
	déploiement de code : ListApplicationRevisions
	déploiement de code : ListApplications
	déploiement de code : ListDeploymentConfigs
	déploiement de code : ListDeploymentGroups
	déploiement de code : ListDeploymentInstances
	déploiement de code : ListDeploymentTargets
	déploiement de code : ListDeployments
	déploiement de code : ListGitHubAccountTokenNames
	déploiement de code : ListOnPremisesInstances
	déploiement de code : PutLifecycleEventHookExecutionStatus
	déploiement de code : RegisterApplicationRevision
	déploiement de code : RegisterOnPremisesInstance
	déploiement de code : SkipWaitTimeForInstanceTermination
	déploiement de code : StopDeployment
	déploiement de code : UpdateApplication
	déploiement de code : UpdateDeploymentGroup

Préfixe de service	Actions
codeguru-profiler	profileur de code : AddNotificationChannels profileur de code : BatchGetFrameMetricData profileur de code : CreateProfilingGroup profileur de code : DeleteProfilingGroup profileur de code : DescribeProfilingGroup profileur de code : GetFindingsReportAccountSummary profileur de code : GetNotificationConfiguration profileur de code : GetPolicy profileur de code : GetProfile profileur de code : GetRecommendations profileur de code : ListFindingsReports profileur de code : ListProfileTimes profileur de code : ListProfilingGroups profileur de code : PutPermission profileur de code : RemoveNotificationChannel profileur de code : RemovePermission profileur de code : SubmitFeedback profileur de code : UpdateProfilingGroup

Préfixe de service	Actions
codeguru-reviewer	réviseur de code : AssociateRepository
	réviseur de code : CreateCodeReview
	réviseur de code : DescribeCodeReview
	réviseur de code : DescribeRecommendationFeedback
	réviseur de code : DescribeRepositoryAssociation
	réviseur de code : DisassociateRepository
	réviseur de code : ListCodeReviews
	réviseur de code : ListRecommendationFeedback
	réviseur de code : ListRecommendations
	réviseur de code : ListRepositoryAssociations
	réviseur de code : PutRecommendationFeedback

Préfixe de service	Actions
codepipeline	pipeline de code : AcknowledgeJob pipeline de code : AcknowledgeThirdPartyJob pipeline de code : CreateCustomActionType pipeline de code : CreatePipeline pipeline de code : DeleteCustomActionType pipeline de code : DeletePipeline pipeline de code : DeleteWebhook pipeline de code : DeregisterWebhookWithThirdParty pipeline de code : GetActionType pipeline de code : GetJobDetails pipeline de code : GetPipeline pipeline de code : GetPipelineExecution pipeline de code : GetPipelineState pipeline de code : GetThirdPartyJobDetails pipeline de code : ListActionExecutions pipeline de code : ListActionTypes pipeline de code : ListPipelineExecutions pipeline de code : ListPipelines pipeline de code : ListRuleExecutions pipeline de code : ListRuleTypes pipeline de code : ListWebhooks

Préfixe de service	Actions
	pipeline de code : OverrideStageCondition pipeline de code : PollForJobs pipeline de code : PollForThirdPartyJobs pipeline de code : PutActionRevision pipeline de code : PutApprovalResult pipeline de code : PutJobFailureResult pipeline de code : PutJobSuccessResult pipeline de code : PutThirdPartyJobFailureResult pipeline de code : PutThirdPartyJobSuccessResult pipeline de code : PutWebhook pipeline de code : RegisterWebhookWithThirdParty pipeline de code : RollbackStage pipeline de code : StartPipelineExecution pipeline de code : StopPipelineExecution pipeline de code : UpdateActionType pipeline de code : UpdatePipeline

Préfixe de service	Actions
codestar	codestar : AssociateTeamMember codestar : CreateProject codestar : CreateUserProfile codestar : DeleteProject codestar : DeleteUserProfile codestar : DescribeProject codestar : DescribeUserProfile codestar : DisassociateTeamMember codestar : ListProjects codestar : ListResources codestar : ListTeamMembers codestar : ListUserProfiles codestar : UpdateProject codestar : UpdateTeamMember codestar : UpdateUserProfile

Préfixe de service	Actions
codestar-notifications	notifications codestar : CreateNotificationRule notifications codestar : DeleteNotificationRule notifications codestar : DeleteTarget notifications codestar : DescribeNotificationRule notifications codestar : ListEventTypes notifications codestar : ListNotificationRules notifications codestar : ListTargets codestar-notifications:Subscribe codestar-notifications:Unsubscribe notifications codestar : UpdateNotificationRule

Préfixe de service	Actions
cognito-identity	identité cognitive : CreateIdentityPool
	identité cognitive : DeleteIdentities
	identité cognitive : DeleteIdentityPool
	identité cognitive : DescribeIdentity
	identité cognitive : DescribeIdentityPool
	identité cognitive : GetIdentityPoolRoles
	identité cognitive : ListIdentities
	identité cognitive : ListIdentityPools
	identité cognitive : LookupDeveloperIdentity
	identité cognitive : MergeDeveloperIdentities
	identité cognitive : SetIdentityPoolRoles
	identité cognitive : UnlinkDeveloperIdentity
	identité cognitive : UpdateIdentityPool

Préfixe de service	Actions
cognito-idp	identifiant : AddCustomAttributes identifiant : AddUserPoolClientSecret identifiant : AdminAddUserToGroup identifiant : AdminConfirmSignUp identifiant : AdminCreateUser identifiant : AdminDeleteUser identifiant : AdminDeleteUserAttributes identifiant : AdminDisableProviderForUser identifiant : AdminDisableUser identifiant : AdminEnableUser identifiant : AdminForgetDevice identifiant : AdminGetDevice identifiant : AdminGetUser identifiant : AdminGetUserAuthFactors identifiant : AdminInitiateAuth identifiant : AdminLinkProviderForUser identifiant : AdminListDevices identifiant : AdminListGroupsForUser identifiant : AdminListUserAuthEvents identifiant : AdminRemoveUserFromGroup identifiant : AdminResetUserPassword

Préfixe de service	Actions
	identifiant : AdminRespondToAuthChallenge identifiant : AdminSetUserMFAPreference identifiant : AdminSetUserPassword identifiant : AdminSetUserSettings identifiant : AdminUpdateAuthEventFeedback identifiant : AdminUpdateDeviceStatus identifiant : AdminUpdateUserAttributes identifiant : AdminUserGlobalSignOut identifiant : AssociateSoftwareToken identifiant : ChangePassword identifiant : ConfirmDevice identifiant : ConfirmForgotPassword identifiant : ConfirmSignUp identifiant : CreateGroup identifiant : CreateIdentityProvider identifiant : CreateManagedLoginBranding identifiant : CreateResourceServer identifiant : CreateTerms identifiant : CreateUserImportJob identifiant : CreateUserPool identifiant : CreateUserPoolClient

Préfixe de service	Actions
	identifiant : CreateUserPoolDomain identifiant : CreateUserPoolReplica identifiant : DeleteGroup identifiant : DeleteIdentityProvider identifiant : DeleteManagedLoginBranding identifiant : DeleteResourceServer identifiant : DeleteTerms identifiant : DeleteUser identifiant : DeleteUserAttributes identifiant : DeleteUserPool identifiant : DeleteUserPoolClient identifiant : DeleteUserPoolClientSecret identifiant : DeleteUserPoolDomain identifiant : DeleteUserPoolReplica identifiant : DescribeIdentityProvider identifiant : DescribeManagedLoginBranding identifiant : DescribeManagedLoginBrandingByClient identifiant : DescribeResourceServer identifiant : DescribeRiskConfiguration identifiant : DescribeTerms identifiant : DescribeUserImportJob

Préfixe de service	Actions
	identifiant : DescribeUserPool identifiant : DescribeUserPoolClient identifiant : DescribeUserPoolDomain identifiant : ForgetDevice identifiant : ForgotPassword cognito-idp:GetCSVHeader identifiant : GetDevice identifiant : GetGroup identifiant : GetIdentityProviderByIdentifier identifiant : GetLogDeliveryConfiguration identifiant : GetProvisionedLimit identifiant : GetSigningCertificate cognito-idp:GetUICustomization identifiant : GetUser identifiant : GetUserAttributeVerificationCode identifiant : GetUserPoolMfaConfig identifiant : GlobalSignOut identifiant : InitiateAuth identifiant : ListDevices identifiant : ListGroups identifiant : ListIdentityProviders

Préfixe de service	Actions
	identifiant : ListResourceServers identifiant : ListTerms identifiant : ListUserImportJobs identifiant : ListUserPoolClientSecrets identifiant : ListUserPoolClients identifiant : ListUserPoolReplicas identifiant : ListUserPools identifiant : ListUsers identifiant : ListUsersInGroup identifiant : ResendConfirmationCode identifiant : RespondToAuthChallenge identifiant : RevokeToken identifiant : SetLogDeliveryConfiguration identifiant : SetRiskConfiguration cognito-idp:SetUICustomization identifiant : SetUserMFAPreference identifiant : SetUserPoolMfaConfig identifiant : SetUserSettings identifiant : SignUp identifiant : StartUserImportJob identifiant : StopUserImportJob

Préfixe de service	Actions
	identifiant : UpdateAuthEventFeedback identifiant : UpdateDeviceStatus identifiant : UpdateGroup identifiant : UpdateIdentityProvider identifiant : UpdateProvisionedLimit identifiant : UpdateResourceServer identifiant : UpdateTerms identifiant : UpdateUserAttributes identifiant : UpdateUserPool identifiant : UpdateUserPoolClient identifiant : UpdateUserPoolDomain identifiant : UpdateUserPoolReplica identifiant : VerifySoftwareToken identifiant : VerifyUserAttribute

Préfixe de service	Actions
cognito-sync	synchronisation cognitive : BulkPublish
	synchronisation cognitive : DeleteDataset
	synchronisation cognitive : DescribeDataset
	synchronisation cognitive : DescribeIdentityPoolUsage
	synchronisation cognitive : DescribeIdentityUsage
	synchronisation cognitive : GetBulkPublishDetails
	synchronisation cognitive : GetCognitoEvents
	synchronisation cognitive : GetIdentityPoolConfiguration
	synchronisation cognitive : ListDatasets
	synchronisation cognitive : ListIdentityPoolUsage
	synchronisation cognitive : ListRecords
	synchronisation cognitive : RegisterDevice
	synchronisation cognitive : SetCognitoEvents
	synchronisation cognitive : SetIdentityPoolConfiguration
	synchronisation cognitive : SubscribeToDataset
	synchronisation cognitive : UnsubscribeFromDataset
	synchronisation cognitive : UpdateRecords

Préfixe de service	Actions
comprehendmedical	médecine complète : DescribeEntitiesDetectionV2Job médecine complète : DescribeICD10CMInferenceJob médecine complète : DescribePHIDetectionJob médecine complète : DescribeRxNormInferenceJob médecine complète : DescribeSNOMEDCTInferenceJob medical complet : V2 DetectEntities comprehendmedical:DetectPHI comprehendmedical:InferICD10CM médecine complète : InferRxNorm comprehendmedical:InferSNOMEDCT médecine complète : ListEntitiesDetectionV2Jobs médecine complète : ListICD10CMInferenceJobs médecine complète : ListPHIDetectionJobs médecine complète : ListRxNormInferenceJobs médecine complète : ListSNOMEDCTInferenceJobs médecine complète : StartEntitiesDetectionV2Job médecine complète : StartICD10CMInferenceJob médecine complète : StartPHIDetectionJob médecine complète : StartRxNormInferenceJob médecine complète : StartSNOMEDCTInferenceJob médecine complète : StopEntitiesDetectionV2Job

Préfixe de service	Actions
	médecine complète : StopICD10CMInferenceJob médecine complète : StopPHIDetectionJob médecine complète : StopRxNormInferenceJob médecine complète : StopSNOMEDCTInferenceJob

Préfixe de service	Actions
compute-optimizer	optimiseur de calcul : DeleteRecommendationPreferences
	optimiseur de calcul : DescribeRecommendationExportJobs
	optimiseur de calcul : ExportAutoScalingGroupRecommendations
	optimiseur de calcul : ExportEBSVolumeRecommendations
	optimiseur de calcul : ExportEC2InstanceRecommendations
	optimiseur de calcul : ExportECSServiceRecommendations
	optimiseur de calcul : ExportIdleRecommendations
	optimiseur de calcul : ExportLambdaFunctionRecommendations
	optimiseur de calcul : ExportLicenseRecommendations
	optimiseur de calcul : ExportRDSDatabaseRecommendations
	optimiseur de calcul : GetEC2RecommendationProjectedMetrics
	optimiseur de calcul : GetECSServiceRecommendationProjectedMetrics
	optimiseur de calcul : GetEffectiveRecommendationPreferences
	optimiseur de calcul : GetEnrollmentStatus
	optimiseur de calcul : GetEnrollmentStatusesForOrganization
	optimiseur de calcul : GetRDSDatabaseRecommendationProjectedMetrics
	optimiseur de calcul : GetRecommendationPreferences
	optimiseur de calcul : GetRecommendationSummaries
	optimiseur de calcul : PutRecommendationPreferences

Préfixe de service	Actions
	optimiseur de calcul : UpdateEnrollmentStatus

Préfixe de service	Actions
config	configuration : BatchGetResourceConfig configuration : DeleteAggregationAuthorization configuration : DeleteConfigRule configuration : DeleteConfigurationAggregator configuration : DeleteConfigurationRecorder configuration : DeleteConformancePack configuration : DeleteDeliveryChannel configuration : DeleteEvaluationResults configuration : DeleteOrganizationConfigRule configuration : DeleteOrganizationConformancePack configuration : DeletePendingAggregationRequest configuration : DeleteRemediationConfiguration configuration : DeleteRemediationExceptions configuration : DeleteResourceConfig configuration : DeleteRetentionConfiguration configuration : DeleteStoredQuery configuration : DeliverConfigSnapshot configuration : DescribeAggregateComplianceByConfigRules configuration : DescribeAggregateComplianceByConformancePacks configuration : DescribeAggregationAuthorizations

Préfixe de service	Actions
	configuration : DescribeComplianceByConfigRule configuration : DescribeComplianceByResource configuration : DescribeConfigRuleEvaluationStatus configuration : DescribeConfigRules configuration : DescribeConfigurationAggregatorSourcesStatus configuration : DescribeConfigurationAggregators configuration : DescribeConfigurationRecorderStatus configuration : DescribeConfigurationRecorders configuration : DescribeConformancePackCompliance configuration : DescribeConformancePackStatus configuration : DescribeConformancePacks configuration : DescribeDeliveryChannelStatus configuration : DescribeDeliveryChannels configuration : DescribeOrganizationConfigRuleStatuses configuration : DescribeOrganizationConfigRules configuration : DescribeOrganizationConformancePackStatuses configuration : DescribeOrganizationConformancePacks configuration : DescribePendingAggregationRequests configuration : DescribeRemediationConfigurations configuration : DescribeRemediationExceptions configuration : DescribeRemediationExecutionStatus

Préfixe de service	Actions
	configuration : DescribeRetentionConfigurations configuration : GetComplianceDetailsByConfigRule configuration : GetComplianceDetailsByResource configuration : GetComplianceSummaryByConfigRule configuration : GetComplianceSummaryByResourceType configuration : GetConformancePackComplianceDetails configuration : GetConformancePackComplianceSummary configuration : GetCustomRulePolicy configuration : GetDiscoveredResourceCounts configuration : GetOrganizationConfigRuleDetailedStatus configuration : GetOrganizationConformancePackDetailedStatus configuration : GetOrganizationCustomRulePolicy configuration : GetResourceConfigHistory configuration : GetResourceEvaluationSummary configuration : GetStoredQuery configuration : ListConfigurationRecorders configuration : ListConformancePackComplianceScores configuration : ListConnectors configuration : ListDiscoveredResources configuration : ListResourceEvaluations configuration : ListStoredQueries

Préfixe de service	Actions
	configuration : PutConfigRule configuration : PutConfigurationAggregator configuration : PutConfigurationRecorder configuration : PutConformancePack configuration : PutConnector configuration : PutDeliveryChannel configuration : PutEvaluations configuration : PutExternalEvaluation configuration : PutOrganizationConfigRule configuration : PutOrganizationConformancePack configuration : PutRemediationConfigurations configuration : PutRemediationExceptions configuration : PutResourceConfig configuration : PutRetentionConfiguration configuration : PutStoredQuery configuration : SelectResourceConfig configuration : StartConfigRulesEvaluation configuration : StartConfigurationRecorder configuration : StartRemediationExecution configuration : StartResourceEvaluation configuration : StopConfigurationRecorder

Préfixe de service	Actions
connexion	connecter : ActivateEvaluationForm
	connecter : AssociateAnalyticsDataSet
	connecter : AssociateApprovedOrigin
	connecter : AssociateBot
	connecter : AssociateContactWithUser
	connecter : AssociateDefaultVocabulary
	connecter : AssociateEmailAddressAlias
	connecter : AssociateFlow
	connecter : AssociateInstanceStorageConfig
	connecter : AssociateLambdaFunction
	connecter : AssociateLexBot
	connecter : AssociatePhoneNumberContactFlow
	connecter : AssociateQueueEmailAddresses
	connecter : AssociateQueueQuickConnects
	connecter : AssociateRoutingProfileQueues
	connecter : AssociateSecurityKey
	connecter : AssociateUserProficiencies
	connecter : BatchAssociateAnalyticsDataSet
	connecter : BatchCreateDataTableValue
	connecter : BatchDeleteDataTableValue
	connecter : BatchDescribeDataTableValue

Préfixe de service	Actions
	connecter : BatchDisassociateAnalyticsDataSet
	connecter : BatchGetFlowAssociation
	connecter : BatchPutContact
	connecter : BatchUpdateDataTableValue
	connecter : ClaimPhoneNumber
	connecter : CreateAgentStatus
	connecter : CreateAttachedFile
	connecter : CreateContact
	connecter : CreateContactFlow
	connecter : CreateContactFlowModule
	connecter : CreateContactFlowModuleAlias
	connecter : CreateContactFlowModuleVersion
	connecter : CreateContactFlowVersion
	connecter : CreateDataTable
	connecter : CreateDataTableAttribute
	connecter : CreateEmailAddress
	connecter : CreateEvaluationForm
	connecter : CreateHoursOfOperation
	connecter : CreateInstance
	connecter : CreateIntegrationAssociation
	connecter : CreateParticipant

Préfixe de service	Actions
	connecter : CreatePersistentContactAssociation
	connecter : CreatePredefinedAttribute
	connecter : CreatePrompt
	connecter : CreatePushNotificationRegistration
	connecter : CreateQueue
	connecter : CreateQuickConnect
	connecter : CreateRoutingProfile
	connecter : CreateRule
	connecter : CreateSecurityProfile
	connecter : CreateTaskTemplate
	connecter : CreateTrafficDistributionGroup
	connecter : CreateUseCase
	connecter : CreateUser
	connecter : CreateUserHierarchyGroup
	connecter : CreateView
	connecter : CreateViewVersion
	connecter : CreateVocabulary
	connecter : CreateWorkspace
	connecter : DeactivateEvaluationForm
	connecter : DeleteContactEvaluation
	connecter : DeleteContactFlow

Préfixe de service	Actions
	connecter : DeleteContactFlowModule
	connecter : DeleteContactFlowModuleAlias
	connecter : DeleteContactFlowModuleVersion
	connecter : DeleteContactFlowVersion
	connecter : DeleteDataTable
	connecter : DeleteDataTableAttribute
	connecter : DeleteEmailAddress
	connecter : DeleteEvaluationForm
	connecter : DeleteHoursOfOperation
	connecter : DeleteHoursOfOperationOverride
	connecter : DeleteInstance
	connecter : DeleteIntegrationAssociation
	connecter : DeleteNotification
	connecter : DeletePredefinedAttribute
	connecter : DeletePrompt
	connecter : DeletePushNotificationRegistration
	connecter : DeleteQueue
	connecter : DeleteQuickConnect
	connecter : DeleteRoutingProfile
	connecter : DeleteRule
	connecter : DeleteSecurityProfile

Préfixe de service	Actions
	connecter : DeleteTaskTemplate
	connecter : DeleteTrafficDistributionGroup
	connecter : DeleteUseCase
	connecter : DeleteUser
	connecter : DeleteUserHierarchyGroup
	connecter : DeleteView
	connecter : DeleteVocabulary
	connecter : DeleteWorkspace
	connecter : DeleteWorkspaceMedia
	connecter : DescribeAttachedFilesConfiguration
	connecter : DescribeAuthenticationProfile
	connecter : DescribeContactFlowModuleAlias
	connecter : DescribeDataTableAttribute
	connecter : DescribeHoursOfOperationOverride
	connecter : DescribeInstanceAttribute
	connecter : DescribeInstanceStorageConfig
	connecter : DescribeNotification
	connecter : DescribePhoneNumber
	connecter : DescribeRule
	connecter : DescribeTrafficDistributionGroup
	connecter : DescribeUserHierarchyStructure

Préfixe de service	Actions
	connecter : DescribeVocabulary
	connecter : DisassociateAnalyticsDataSet
	connecter : DisassociateApprovedOrigin
	connecter : DisassociateBot
	connecter : DisassociateEmailAddressAlias
	connecter : DisassociateFlow
	connecter : DisassociateInstanceStorageConfig
	connecter : DisassociateLambdaFunction
	connecter : DisassociateLexBot
	connecter : DisassociatePhoneNumberContactFlow
	connecter : DisassociateQueueEmailAddresses
	connecter : DisassociateQueueQuickConnects
	connecter : DisassociateRoutingProfileQueues
	connecter : DisassociateSecurityKey
	connecter : DisassociateUserProficiencies
	connecter : DismissUserContact
	connecter : EvaluateDataTableValues
	connecter : GetContactAttributes
	connecter : GetContactMetrics
	connecter : GetCurrentMetricData
	connecter : GetCurrentUserData

Préfixe de service	Actions
	connecter : GetEffectiveHoursOfOperations
	connecter : GetEvaluationFormValidation
	connecter : GetFederationToken
	connecter : GetFlowAssociation
	connecter : GetMetricData
	connecter : GetMetricDataV2
	connecter : GetPromptFile
	connecter : GetTaskTemplate
	connecter : GetTrafficDistribution
	connecter : ImportPhoneNumber
	connecter : ImportWorkspaceMedia
	connecter : ListAnalyticsDataAssociations
	connecter : ListAnalyticsDataLakeDataSets
	connecter : ListApprovedOrigins
	connecter : ListAssociatedContacts
	connecter : ListAttachedFilesConfigurations
	connecter : ListAuthenticationProfiles
	connecter : ListBots
	connecter : ListContactEvaluations
	connecter : ListContactFlowModuleAliases
	connecter : ListContactFlowModuleVersions

Préfixe de service	Actions
	connecter : ListContactFlowModules
	connecter : ListContactFlowVersions
	connecter : ListContactFlows
	connecter : ListContactReferences
	connecter : ListDataTableAttributes
	connecter : ListDataTablePrimaryValues
	connecter : ListDataTableValues
	connecter : ListDataTables
	connecter : ListDefaultVocabularies
	connecter : ListEvaluationFormVersions
	connecter : ListEvaluationForms
	connecter : ListFlowAssociations
	connecter : ListHoursOfOperations
	connecter : ListInstanceAttributes
	connecter : ListInstanceStorageConfigs
	connecter : ListIntegrationAssociations
	connecter : ListLambdaFunctions
	connecter : ListLexBots
	connecter : ListNotifications
	connecter : ListPhoneNumbers
	connecter : ListPhoneNumbersV2

Préfixe de service	Actions
	connecter : ListPredefinedAttributes
	connecter : ListPrompts
	connecter : ListQueueEmailAddresses
	connecter : ListQueueQuickConnects
	connecter : ListQueues
	connecter : ListQuickConnects
	connecter : ListRealtimeContactAnalysisSegmentsV2
	connecter : ListRoutingProfileManualAssignmentQueues
	connecter : ListRoutingProfileQueues
	connecter : ListRoutingProfiles
	connecter : ListRules
	connecter : ListSecurityKeys
	connecter : ListSecurityProfileApplications
	connecter : ListSecurityProfileFlowModules
	connecter : ListSecurityProfilePermissions
	connecter : ListSecurityProfiles
	connecter : ListTaskTemplates
	connecter : ListTrafficDistributionGroups
	connecter : ListUseCases
	connecter : ListUserHierarchyGroups
	connecter : ListUserNotifications

Préfixe de service	Actions
	connecter : ListUsers
	connecter : ListViewVersions
	connecter : ListViews
	connecter : ListWorkspaceMedia
	connecter : ListWorkspacePages
	connecter : ListWorkspaces
	connecter : MonitorContact
	connecter : PauseContact
	connecter : PutUserStatus
	connecter : ReleasePhoneNumber
	connecter : ReplicateInstance
	connecter : ResumeContact
	connecter : ResumeContactRecording
	connecter : SearchAgentStatuses
	connecter : SearchAvailablePhoneNumbers
	connecter : SearchContactEvaluations
	connecter : SearchContactFlowModules
	connecter : SearchContactFlows
	connecter : SearchContacts
	connecter : SearchDataTables
	connecter : SearchEmailAddresses

Préfixe de service	Actions
	connecter : SearchEvaluationForms
	connecter : SearchHoursOfOperations
	connecter : SearchPredefinedAttributes
	connecter : SearchPrompts
	connecter : SearchQueues
	connecter : SearchQuickConnects
	connecter : SearchRoutingProfiles
	connecter : SearchRules
	connecter : SearchSecurityProfiles
	connecter : SearchUserHierarchyGroups
	connecter : SearchViews
	connecter : SearchVocabularies
	connecter : SearchWorkspaceAssociations
	connecter : SearchWorkspaces
	connecter : SendChatIntegrationEvent
	connecter : SendOutboundEmail
	connecter : StartChatContact
	connecter : StartContactConversationalAnalyticsJob
	connecter : StartContactEvaluation
	connecter : StartContactMediaProcessing
	connecter : StartContactRecording

Préfixe de service	Actions
	connecter : StartContactStreaming
	connecter : StartEmailContact
	connecter : StartOutboundChatContact
	connecter : StartOutboundEmailContact
	connecter : StartOutboundVoiceContact
	connecter : StartScreenSharing
	connecter : StartTaskContact
	connecter : StartWebRTCContact
	connecter : StopContact
	connecter : StopContactMediaProcessing
	connecter : StopContactRecording
	connecter : StopContactStreaming
	connecter : SubmitContactEvaluation
	connecter : SuspendContactRecording
	connecter : TransferContact
	connecter : UpdateAgentStatus
	connecter : UpdateAttachedFilesConfiguration
	connecter : UpdateAuthenticationProfile
	connecter : UpdateContact
	connecter : UpdateContactAttributes
	connecter : UpdateContactEvaluation

Préfixe de service	Actions
	connecter : UpdateContactFlowContent
	connecter : UpdateContactFlowMetadata
	connecter : UpdateContactFlowModuleAlias
	connecter : UpdateContactFlowModuleContent
	connecter : UpdateContactFlowModuleMetadata
	connecter : UpdateContactFlowName
	connecter : UpdateContactRoutingData
	connecter : UpdateContactSchedule
	connecter : UpdateDataTableAttribute
	connecter : UpdateDataTableMetadata
	connecter : UpdateDataTablePrimaryValues
	connecter : UpdateEmailAddressMetadata
	connecter : UpdateEvaluationForm
	connecter : UpdateHoursOfOperation
	connecter : UpdateHoursOfOperationOverride
	connecter : UpdateInstanceAttribute
	connecter : UpdateInstanceStorageConfig
	connecter : UpdateNotificationContent
	connecter : UpdateParticipantAuthentication
	connecter : UpdateParticipantRoleConfig
	connecter : UpdatePhoneNumber

Préfixe de service	Actions
	connecter : UpdatePhoneNumberMetadata
	connecter : UpdatePredefinedAttribute
	connecter : UpdatePrompt
	connecter : UpdateQueueHoursOfOperation
	connecter : UpdateQueueMaxContacts
	connecter : UpdateQueueName
	connecter : UpdateQueueOutboundCallerConfig
	connecter : UpdateQueueOutboundEmailConfig
	connecter : UpdateQueueStatus
	connecter : UpdateQuickConnectConfig
	connecter : UpdateQuickConnectName
	connecter : UpdateRoutingProfileAgentAvailabilityTimer
	connecter : UpdateRoutingProfileConcurrency
	connecter : UpdateRoutingProfileDefaultOutboundQueue
	connecter : UpdateRoutingProfileName
	connecter : UpdateRoutingProfileQueues
	connecter : UpdateRule
	connecter : UpdateSecurityProfile
	connecter : UpdateTaskTemplate
	connecter : UpdateTrafficDistribution
	connecter : UpdateUserHierarchy

Préfixe de service	Actions
	connecter : UpdateUserHierarchyGroupName
	connecter : UpdateUserHierarchyStructure
	connecter : UpdateUserIdentityInfo
	connecter : UpdateUserNotificationStatus
	connecter : UpdateUserPhoneConfig
	connecter : UpdateUserProficiencies
	connecter : UpdateUserRoutingProfile
	connecter : UpdateUserSecurityProfiles
	connecter : UpdateViewContent
	connecter : UpdateViewMetadata
	connecter : UpdateWorkspaceMetadata
	connecter : UpdateWorkspaceTheme
	connecter : UpdateWorkspaceVisibility
cur	cœur : DeleteReportDefinition
	cœur : DescribeReportDefinitions
	cœur : ModifyReportDefinition
	cœur : PutReportDefinition

Préfixe de service	Actions
databrew	base de données : BatchDeleteRecipeVersion
	base de données : CreateDataset
	base de données : CreateProfileJob
	base de données : CreateProject
	base de données : CreateRecipe
	base de données : CreateRecipeJob
	base de données : CreateRuleset
	base de données : CreateSchedule
	base de données : DeleteDataset
	base de données : DeleteJob
	base de données : DeleteProject
	base de données : DeleteRecipeVersion
	base de données : DeleteRuleset
	base de données : DeleteSchedule
	base de données : DescribeDataset
	base de données : DescribeJob
	base de données : DescribeJobRun
	base de données : DescribeProject
	base de données : DescribeRecipe
	base de données : DescribeRuleset
	base de données : DescribeSchedule

Préfixe de service	Actions
	base de données : ListDatasets
	base de données : ListJobRuns
	base de données : ListJobs
	base de données : ListProjects
	base de données : ListRecipeVersions
	base de données : ListRecipes
	base de données : ListRulesets
	base de données : ListSchedules
	base de données : PublishRecipe
	base de données : SendProjectSessionAction
	base de données : StartJobRun
	base de données : StartProjectSession
	base de données : StopJobRun
	base de données : UpdateDataset
	base de données : UpdateProfileJob
	base de données : UpdateProject
	base de données : UpdateRecipe
	base de données : UpdateRecipeJob
	base de données : UpdateRuleset
	base de données : UpdateSchedule

Préfixe de service	Actions
dataexchange	échange de données : AcceptDataGrant échange de données : CancelJob échange de données : CreateDataGrant échange de données : CreateDataSet échange de données : CreateEventAction échange de données : CreateJob échange de données : CreateRevision échange de données : DeleteAsset échange de données : DeleteDataGrant échange de données : DeleteEventAction échange de données : DeleteRevision échange de données : GetDataGrant échange de données : GetEventAction échange de données : GetJob échange de données : GetReceivedDataGrant échange de données : ListDataGrants échange de données : ListDataSetRevisions échange de données : ListDataSets échange de données : ListEventActions échange de données : ListJobs échange de données : ListReceivedDataGrants

Préfixe de service	Actions
	échange de données : ListRevisionAssets échange de données : RevokeRevision échange de données : SendDataSetNotification échange de données : StartJob échange de données : UpdateAsset échange de données : UpdateDataSet échange de données : UpdateEventAction échange de données : UpdateRevision

Préfixe de service	Actions
datapipeline	pipeline de données : ActivatePipeline pipeline de données : CreatePipeline pipeline de données : DeactivatePipeline pipeline de données : DeletePipeline pipeline de données : DescribeObjects pipeline de données : DescribePipelines pipeline de données : EvaluateExpression pipeline de données : GetPipelineDefinition pipeline de données : ListPipelines pipeline de données : PollForTask pipeline de données : PutPipelineDefinition pipeline de données : QueryObjects pipeline de données : ReportTaskProgress pipeline de données : ReportTaskRunnerHeartbeat pipeline de données : SetStatus pipeline de données : SetTaskStatus pipeline de données : ValidatePipelineDefinition

Préfixe de service	Actions
dax	télécopieur : CreateCluster télécopieur : DecreaseReplicationFactor télécopieur : DeleteCluster télécopieur : DeleteParameterGroup télécopieur : DeleteSubnetGroup télécopieur : DescribeClusters télécopieur : DescribeDefaultParameters télécopieur : DescribeEvents télécopieur : DescribeParameterGroups télécopieur : DescribeParameters télécopieur : DescribeSubnetGroups télécopieur : IncreaseReplicationFactor télécopieur : RebootNode télécopieur : UpdateCluster télécopieur : UpdateParameterGroup télécopieur : UpdateSubnetGroup

Préfixe de service	Actions
devicefarm	ferme d'appareils : CreateDevicePool
	ferme d'appareils : CreateInstanceProfile
	ferme d'appareils : CreateNetworkProfile
	ferme d'appareils : CreateProject
	ferme d'appareils : CreateRemoteAccessSession
	ferme d'appareils : CreateTestGridProject
	ferme d'appareils : CreateTestGridUrl
	ferme d'appareils : CreateUpload
	devicefarm:CreateVPCEConfiguration
	ferme d'appareils : DeleteDevicePool
	ferme d'appareils : DeleteInstanceProfile
	ferme d'appareils : DeleteNetworkProfile
	ferme d'appareils : DeleteProject
	ferme d'appareils : DeleteRemoteAccessSession
	ferme d'appareils : DeleteRun
	ferme d'appareils : DeleteTestGridProject
	ferme d'appareils : DeleteUpload
	devicefarm:DeleteVPCEConfiguration
	ferme d'appareils : GetAccountSettings
	ferme d'appareils : GetDevice
	ferme d'appareils : GetDeviceInstance

Préfixe de service	Actions
	ferme d'appareils : GetDevicePool
	ferme d'appareils : GetDevicePoolCompatibility
	ferme d'appareils : GetInstanceProfile
	ferme d'appareils : GetJob
	ferme d'appareils : GetNetworkProfile
	ferme d'appareils : GetOfferingStatus
	ferme d'appareils : GetProject
	ferme d'appareils : GetRemoteAccessSession
	ferme d'appareils : GetRun
	ferme d'appareils : GetSuite
	ferme d'appareils : GetTest
	ferme d'appareils : GetTestGridProject
	ferme d'appareils : GetTestGridSession
	ferme d'appareils : GetUpload
	devicefarm:GetVPCEConfiguration
	ferme d'appareils : ListArtifacts
	ferme d'appareils : ListDeviceInstances
	ferme d'appareils : ListDevicePools
	ferme d'appareils : ListDevices
	ferme d'appareils : ListInstanceProfiles
	ferme d'appareils : ListJobs

Préfixe de service	Actions
	ferme d'appareils : ListNetworkProfiles
	ferme d'appareils : ListOfferingPromotions
	ferme d'appareils : ListOfferingTransactions
	ferme d'appareils : ListOfferings
	ferme d'appareils : ListProjects
	ferme d'appareils : ListRemoteAccessSessions
	ferme d'appareils : ListRuns
	ferme d'appareils : ListSamples
	ferme d'appareils : ListSuites
	ferme d'appareils : ListTestGridProjects
	ferme d'appareils : ListTestGridSessionActions
	ferme d'appareils : ListTestGridSessionArtifacts
	ferme d'appareils : ListTestGridSessions
	ferme d'appareils : ListTests
	ferme d'appareils : ListUniqueProblems
	ferme d'appareils : ListUploads
	devicefarm:ListVPCEConfigurations
	ferme d'appareils : PurchaseOffering
	ferme d'appareils : RenewOffering
	ferme d'appareils : ScheduleRun
	ferme d'appareils : StopJob

Préfixe de service	Actions
	ferme d'appareils : StopRemoteAccessSession
	ferme d'appareils : StopRun
	ferme d'appareils : UpdateDeviceInstance
	ferme d'appareils : UpdateDevicePool
	ferme d'appareils : UpdateInstanceProfile
	ferme d'appareils : UpdateNetworkProfile
	ferme d'appareils : UpdateProject
	ferme d'appareils : UpdateTestGridProject
	ferme d'appareils : UpdateUpload
	devicefarm:UpdateVPCEConfiguration

Préfixe de service	Actions
devops-guru	gourou de DevOps : AddNotificationChannel gourou de DevOps : DeleteInsight gourou de DevOps : DescribeAccountHealth gourou de DevOps : DescribeAccountOverview gourou de DevOps : DescribeAnomaly gourou de DevOps : DescribeEventSourcesConfig gourou de DevOps : DescribeFeedback gourou de DevOps : DescribeInsight gourou de DevOps : DescribeOrganizationHealth gourou de DevOps : DescribeOrganizationOverview gourou de DevOps : DescribeOrganizationResourceCollectionHealth gourou de DevOps : DescribeResourceCollectionHealth gourou de DevOps : DescribeServiceIntegration gourou de DevOps : GetCostEstimation gourou de DevOps : GetResourceCollection gourou de DevOps : ListAnomaliesForInsight gourou de DevOps : ListAnomalousLogGroups gourou de DevOps : ListEvents gourou de DevOps : ListInsights gourou de DevOps : ListMonitoredResources

Préfixe de service	Actions
	gourou de DevOps : ListNotificationChannels gourou de DevOps : ListOrganizationInsights gourou de DevOps : ListRecommendations gourou de DevOps : PutFeedback gourou de DevOps : RemoveNotificationChannel gourou de DevOps : SearchInsights gourou de DevOps : SearchOrganizationInsights gourou de DevOps : StartCostEstimation gourou de DevOps : UpdateEventSourcesConfig gourou de DevOps : UpdateResourceCollection gourou de DevOps : UpdateServiceIntegration

Préfixe de service	Actions
directconnect	connexion directe : AcceptDirectConnectGatewayAssociationProposal
	connexion directe : AllocateConnectionOnInterconnect
	connexion directe : AllocateHostedConnection
	connexion directe : AllocatePrivateVirtualInterface
	connexion directe : AllocatePublicVirtualInterface
	connexion directe : AllocateTransitVirtualInterface
	connexion directe : AssociateConnectionWithLag
	connexion directe : AssociateHostedConnection
	connexion directe : AssociateMacSecKey
	connexion directe : AssociateVirtualInterface
	connexion directe : ConfirmConnection
	connexion directe : ConfirmCustomerAgreement
	connexion directe : ConfirmPrivateVirtualInterface
	connexion directe : ConfirmPublicVirtualInterface
	connexion directe : ConfirmTransitVirtualInterface
	directconnect:CreateBGPPeer
	connexion directe : CreateConnection
	connexion directe : CreateDirectConnectGateway
	connexion directe : CreateDirectConnectGatewayAssociation
	connexion directe : CreateDirectConnectGatewayAssociationProposal

Préfixe de service	Actions
	connexion directe : CreateInterconnect
	connexion directe : CreateLag
	connexion directe : CreatePrivateVirtualInterface
	connexion directe : CreatePublicVirtualInterface
	connexion directe : CreateTransitVirtualInterface
	directconnect:DeleteBGPPeer
	connexion directe : DeleteConnection
	connexion directe : DeleteDirectConnectGateway
	connexion directe : DeleteDirectConnectGatewayAssociation
	connexion directe : DeleteDirectConnectGatewayAssociationProposal
	connexion directe : DeleteInterconnect
	connexion directe : DeleteLag
	connexion directe : DeleteVirtualInterface
	connexion directe : DescribeConnectionLoa
	connexion directe : DescribeConnections
	connexion directe : DescribeConnectionsOnInterconnect
	connexion directe : DescribeCustomerMetadata
	connexion directe : DescribeDirectConnectGatewayAssociationProposals
	connexion directe : DescribeDirectConnectGatewayAssociations
	connexion directe : DescribeDirectConnectGatewayAttachments

Préfixe de service	Actions
	connexion directe : DescribeDirectConnectGateways
	connexion directe : DescribeHostedConnections
	connexion directe : DescribeInterconnectLoa
	connexion directe : DescribeInterconnects
	connexion directe : DescribeLags
	connexion directe : DescribeLoa
	connexion directe : DescribeLocations
	connexion directe : DescribeRouterConfiguration
	connexion directe : DescribeVirtualGateways
	connexion directe : DescribeVirtualInterfaces
	connexion directe : DisassociateConnectionFromLag
	connexion directe : DisassociateMacSecKey
	connexion directe : ListVirtualInterfaceTestHistory
	connexion directe : StartBgpFailoverTest
	connexion directe : StopBgpFailoverTest
	connexion directe : UpdateConnection
	connexion directe : UpdateDirectConnectGateway
	connexion directe : UpdateDirectConnectGatewayAssociation
	connexion directe : UpdateLag
	connexion directe : UpdateVirtualInterfaceAttributes

Préfixe de service	Actions
dlm	DLM : CreateLifecyclePolicy DLM : DeleteLifecyclePolicy DLM : GetLifecyclePolicies DLM : GetLifecyclePolicy DLM : UpdateLifecyclePolicy

Préfixe de service	Actions
dms	dms : ApplyPendingMaintenanceAction
	dms : AssociateExtensionPack
	dms : BatchStartRecommendations
	dms : CancelMetadataModelCreation
	dms : CancelReplicationTaskAssessmentRun
	dms : CreateDataProvider
	dms : CreateEndpoint
	dms : CreateEventSubscription
	dms : CreateInstanceProfile
	dms : CreateMigrationProject
	dms : CreateReplicationConfig
	dms : CreateReplicationInstance
	dms : CreateReplicationSubnetGroup
	dms : CreateReplicationTask
	dms : DeleteCertificate
	dms : DeleteConnection
	dms : DeleteDataMigration
	dms : DeleteDataProvider
	dms : DeleteEndpoint
	dms : DeleteEventSubscription
	dms : DeleteFleetAdvisorCollector

Préfixe de service	Actions
	dms : DeleteFleetAdvisorDatabases
	dms : DeleteInstanceProfile
	dms : DeleteMigrationProject
	dms : DeleteReplicationConfig
	dms : DeleteReplicationInstance
	dms : DeleteReplicationSubnetGroup
	dms : DeleteReplicationTask
	dms : DeleteReplicationTaskAssessmentRun
	dms : DescribeAccountAttributes
	dms : DescribeApplicableIndividualAssessments
	dms : DescribeCertificates
	dms : DescribeConnections
	dms : DescribeDataMigrations
	dms : DescribeEndpointSettings
	dms : DescribeEndpointTypes
	dms : DescribeEndpoints
	dms : DescribeEngineVersions
	dms : DescribeEventCategories
	dms : DescribeEventSubscriptions
	dms : DescribeEvents
	dms : DescribeFleetAdvisorCollectors

Préfixe de service	Actions
	dms : DescribeFleetAdvisorDatabases
	dms : DescribeFleetAdvisorLsaAnalysis
	dms : DescribeFleetAdvisorSchemaObjectSummary
	dms : DescribeFleetAdvisorSchemas
	dms : DescribeMetadataModel
	dms : DescribeMetadataModelChildren
	dms : DescribeMetadataModelCreations
	dms : DescribeMetadataModelImports
	dms : DescribeOrderableReplicationInstances
	dms : DescribePendingMaintenanceActions
	dms : DescribeRecommendationLimitations
	dms : DescribeRecommendations
	dms : DescribeRefreshSchemasStatus
	dms : DescribeReplicationConfigs
	dms : DescribeReplicationInstanceTaskLogs
	dms : DescribeReplicationInstances
	dms : DescribeReplicationSubnetGroups
	dms : DescribeReplicationTableStatistics
	dms : DescribeReplicationTaskAssessmentResults
	dms : DescribeReplicationTaskAssessmentRuns
	dms : DescribeReplicationTaskIndividualAssessments

Préfixe de service	Actions
	dms : DescribeReplicationTasks
	dms : DescribeReplications
	dms : DescribeSchemas
	dms : DescribeTableStatistics
	dms : ExportMetadataModelAssessment
	dms : ImportCertificate
	dms : ListDataProviders
	dms : ListExtensionPacks
	dms : ListInstanceProfiles
	dms : ListMetadataModelAssessments
	dms : ListMetadataModelConversions
	dms : ListMetadataModelExports
	dms : ListMigrationProjects
	dms : ModifyDataMigration
	dms : ModifyEndpoint
	dms : ModifyEventSubscription
	dms : ModifyReplicationConfig
	dms : ModifyReplicationInstance
	dms : ModifyReplicationSubnetGroup
	dms : ModifyReplicationTask
	dms : MoveReplicationTask

Préfixe de service	Actions
	dms : RebootReplicationInstance
	dms : RefreshSchemas
	dms : ReloadReplicationTables
	dms : ReloadTables
	dms : RunFleetAdvisorLsaAnalysis
	dms : StartMetadataModelAssessment
	dms : StartMetadataModelConversion
	dms : StartMetadataModelCreation
	dms : StartMetadataModelExportAsScripts
	dms : StartMetadataModelExportToTarget
	dms : StartRecommendations
	dms : StartReplication
	dms : StartReplicationTask
	dms : StartReplicationTaskAssessment
	dms : StopDataMigration
	dms : StopReplicationTask
	dms : TestConnection
	dms : UpdateConversionConfiguration
	dms : UpdateDataProvider
	dms : UpdateInstanceProfile
	dms : UpdateMigrationProject

Préfixe de service	Actions
	dms : UpdateSubscriptionsToEventBridge
docdb-elastic	docdb-elastic : ApplyPendingMaintenanceAction docdb-elastic : CopyClusterSnapshot docdb-elastic : DeleteCluster docdb-elastic : DeleteClusterSnapshot docdb-elastic : GetCluster docdb-elastic : GetClusterSnapshot docdb-elastic : GetPendingMaintenanceAction docdb-elastic : ListClusterSnapshots docdb-elastic : ListClusters docdb-elastic : ListPendingMaintenanceActions docdb-elastic : RestoreClusterFromSnapshot docdb-elastic : StartCluster docdb-elastic : StopCluster docdb-elastic : UpdateCluster

Préfixe de service	Actions
dynamodb	dynamodb : CreateBackup
	dynamodb : CreateGlobalTable
	dynamodb : CreateTable
	dynamodb : DeleteBackup
	dynamodb : DeleteTable
	dynamodb : DescribeBackup
	dynamodb : DescribeContinuousBackups
	dynamodb : DescribeContributorInsights
	dynamodb : DescribeEndpoints
	dynamodb : DescribeExport
	dynamodb : DescribeGlobalTable
	dynamodb : DescribeGlobalTableSettings
	dynamodb : DescribeImport
	dynamodb : DescribeKinesisStreamingDestination
	dynamodb : DescribeLimits
	dynamodb : DescribeStream
	dynamodb : DescribeTable
	dynamodb : DescribeTableReplicaAutoScaling
	dynamodb : DescribeTimeToLive
	dynamodb : DisableKinesisStreamingDestination
	dynamodb : EnableKinesisStreamingDestination

Préfixe de service	Actions
	dynamodb : ExportTableToPointInTime
	dynamodb : GetResourcePolicy
	dynamodb : ImportTable
	dynamodb : ListBackups
	dynamodb : ListContributorInsights
	dynamodb : ListExports
	dynamodb : ListGlobalTables
	dynamodb : ListImports
	dynamodb : ListStreams
	dynamodb : ListTables
	dynamodb : RestoreTableFromBackup
	dynamodb : RestoreTableToPointInTime
	dynamodb : UpdateContinuousBackups
	dynamodb : UpdateContributorInsights
	dynamodb : UpdateGlobalTable
	dynamodb : UpdateGlobalTableSettings
	dynamodb : UpdateKinesisStreamingDestination
	dynamodb : UpdateTable
	dynamodb : UpdateTableReplicaAutoScaling
	dynamodb : UpdateTimeToLive

Préfixe de service	Actions
ebs	ebs : CompleteSnapshot ebs : StartSnapshot

Préfixe de service	Actions
ec2	EC2 : AcceptAddressTransfer
	EC2 : AcceptCapacityReservationBillingOwnership
	EC2 : AcceptReservedInstancesExchangeQuote
	EC2 : AcceptTransitGatewayClientVpnAttachment
	EC2 : AcceptTransitGatewayMulticastDomainAssociations
	EC2 : AcceptTransitGatewayPeeringAttachment
	EC2 : AcceptTransitGatewayVpcAttachment
	EC2 : AcceptVpcEndpointConnections
	EC2 : AcceptVpcPeeringConnection
	EC2 : AdvertiseByoipCidr
	EC2 : AllocateAddress
	EC2 : AllocateHosts
	EC2 : AllocateIpamPoolCidr
	EC2 : ApplySecurityGroupsToClientVpnTargetNetwork
	EC2 : AssignIpv6Addresses
	EC2 : AssignPrivateIpAddresses
	EC2 : AssignPrivateNatGatewayAddress
	EC2 : AssociateAddress
	EC2 : AssociateCapacityReservationBillingOwner
	EC2 : AssociateClientVpnTargetNetwork
	EC2 : AssociateDhcpOptions

Préfixe de service	Actions
	EC2 : AssociateEnclaveCertificateIamRole
	EC2 : AssociateIamInstanceProfile
	EC2 : AssociateInstanceEventWindow
	EC2 : AssociateIpamByoasn
	EC2 : AssociateIpamResourceDiscovery
	EC2 : AssociateNatGatewayAddress
	EC2 : AssociateRouteServer
	EC2 : AssociateRouteTable
	EC2 : AssociateSecurityGroupVpc
	EC2 : AssociateSubnetCidrBlock
	EC2 : AssociateTransitGatewayMulticastDomain
	EC2 : AssociateTransitGatewayPolicyTable
	EC2 : AssociateTransitGatewayRouteTable
	EC2 : AssociateTrunkInterface
	EC2 : AssociateVpcCidrBlock
	EC2 : AttachClassicLinkVpc
	EC2 : AttachImageWatermark
	EC2 : AttachInternetGateway
	EC2 : AttachNetworkInterface
	EC2 : AttachVerifiedAccessTrustProvider
	EC2 : AttachVolume

Préfixe de service	Actions
	EC2 : AttachVpnGateway
	EC2 : AuthorizeClientVpnIngress
	EC2 : AuthorizeSecurityGroupEgress
	EC2 : AuthorizeSecurityGroupIngress
	EC2 : BundleInstance
	EC2 : CancelBundleTask
	EC2 : CancelCapacityReservation
	EC2 : CancelCapacityReservationFleets
	EC2 : CancelConversionTask
	EC2 : CancelDeclarativePoliciesReport
	EC2 : CancelExportTask
	EC2 : CancelImageLaunchPermission
	EC2 : CancelImportTask
	EC2 : CancelReservedInstancesListing
	EC2 : CancelSpotFleetRequests
	EC2 : CancelSpotInstanceRequests
	EC2 : ConfirmProductInstance
	EC2 : CopyFpgaImage
	EC2 : CopyImage
	EC2 : CopySnapshot
	EC2 : CopyVolumes

Préfixe de service	Actions
	EC2 : CreateCapacityManagerDataExport
	EC2 : CreateCapacityReservation
	EC2 : CreateCapacityReservationBySplitting
	EC2 : CreateCapacityReservationCancellationQuote
	EC2 : CreateCapacityReservationFleet
	EC2 : CreateCarrierGateway
	EC2 : CreateClientVpnEndpoint
	EC2 : CreateClientVpnRoute
	EC2 : CreateCoipCidr
	EC2 : CreateCoipPool
	EC2 : CreateCustomerGateway
	EC2 : CreateDefaultSubnet
	EC2 : CreateDefaultVpc
	EC2 : CreateDelegateMacVolumeOwnershipTask
	EC2 : CreateDhcpOptions
	EC2 : CreateEgressOnlyInternetGateway
	EC2 : CreateFleet
	EC2 : CreateFlowLogs
	EC2 : CreateFpgaImage
	EC2 : CreateImage
	EC2 : CreateImageUsageReport

Préfixe de service	Actions
	EC2 : CreateInstanceConnectEndpoint
	EC2 : CreateInstanceEventWindow
	EC2 : CreateInstanceExportTask
	EC2 : CreateInternetGateway
	EC2 : CreateInterruptibleCapacityReservationAllocation
	EC2 : CreateIpam
	EC2 : CreateIpamExternalResourceVerificationToken
	EC2 : CreateIpamPolicy
	EC2 : CreateIpamPool
	EC2 : CreateIpamPrefixListResolver
	EC2 : CreateIpamPrefixListResolverTarget
	EC2 : CreateIpamResourceDiscovery
	EC2 : CreateIpamScope
	EC2 : CreateKeyPair
	EC2 : CreateLaunchTemplateVersion
	EC2 : CreateLocalGatewayRoute
	EC2 : CreateLocalGatewayRouteTable
	EC2 : CreateLocalGatewayRouteTableVirtualInterfaceGroupAssoc iation
	EC2 : CreateLocalGatewayRouteTableVpcAssociation
	EC2 : CreateLocalGatewayVirtualInterface

Préfixe de service	Actions
	EC2 : CreateLocalGatewayVirtualInterfaceGroup
	EC2 : CreateMacSystemIntegrityProtectionModificationTask
	EC2 : CreateManagedPrefixList
	EC2 : CreateNatGateway
	EC2 : CreateNetworkAcl
	EC2 : CreateNetworkAclEntry
	EC2 : CreateNetworkInsightsAccessScope
	EC2 : CreateNetworkInsightsPath
	EC2 : CreateNetworkInterface
	EC2 : CreateNetworkInterfacePermission
	EC2 : CreatePlacementGroup
	EC2 : CreatePublicIpv4Pool
	EC2 : CreateReplaceRootVolumeTask
	EC2 : CreateReservedInstancesListing
	EC2 : CreateRestoreImageTask
	EC2 : CreateRoute
	EC2 : CreateRouteServer
	EC2 : CreateRouteServerEndpoint
	EC2 : CreateRouteServerPeer
	EC2 : CreateRouteTable
	EC2 : CreateSecondaryNetwork

Préfixe de service	Actions
	EC2 : CreateSecondarySubnet
	EC2 : CreateSecurityGroup
	EC2 : CreateSnapshots
	EC2 : CreateSpotDatafeedSubscription
	EC2 : CreateStoreImageTask
	EC2 : CreateSubnet
	EC2 : CreateSubnetCidrReservation
	EC2 : CreateTrafficMirrorFilter
	EC2 : CreateTrafficMirrorFilterRule
	EC2 : CreateTrafficMirrorSession
	EC2 : CreateTrafficMirrorTarget
	EC2 : CreateTransitGateway
	EC2 : CreateTransitGatewayConnect
	EC2 : CreateTransitGatewayConnectPeer
	EC2 : CreateTransitGatewayMeteringPolicy
	EC2 : CreateTransitGatewayMeteringPolicyEntry
	EC2 : CreateTransitGatewayMulticastDomain
	EC2 : CreateTransitGatewayPeeringAttachment
	EC2 : CreateTransitGatewayPolicyTable
	EC2 : CreateTransitGatewayPrefixListReference
	EC2 : CreateTransitGatewayRoute

Préfixe de service	Actions
	EC2 : CreateTransitGatewayRouteTable
	EC2 : CreateTransitGatewayRouteTableAnnouncement
	EC2 : CreateTransitGatewayVpcAttachment
	EC2 : CreateVerifiedAccessEndpoint
	EC2 : CreateVerifiedAccessGroup
	EC2 : CreateVerifiedAccessInstance
	EC2 : CreateVerifiedAccessTrustProvider
	EC2 : CreateVolume
	EC2 : CreateVpc
	EC2 : CreateVpcBlockPublicAccessExclusion
	EC2 : CreateVpcEncryptionControl
	EC2 : CreateVpcEndpoint
	EC2 : CreateVpcEndpointConnectionNotification
	EC2 : CreateVpcEndpointServiceConfiguration
	EC2 : CreateVpcPeeringConnection
	EC2 : CreateVpnConcentrator
	EC2 : CreateVpnConnection
	EC2 : CreateVpnConnectionRoute
	EC2 : CreateVpnGateway
	EC2 : DeleteCapacityManagerDataExport
	EC2 : DeleteCarrierGateway

Préfixe de service	Actions
	EC2 : DeleteClientVpnEndpoint
	EC2 : DeleteClientVpnRoute
	EC2 : DeleteCoipCidr
	EC2 : DeleteCoipPool
	EC2 : DeleteCustomerGateway
	EC2 : DeleteDhcpOptions
	EC2 : DeleteEgressOnlyInternetGateway
	EC2 : DeleteFleets
	EC2 : DeleteFlowLogs
	EC2 : DeleteFpgaImage
	EC2 : DeleteImageUsageReport
	EC2 : DeleteInstanceConnectEndpoint
	EC2 : DeleteInstanceEventWindow
	EC2 : DeleteInternetGateway
	EC2 : DeleteIpam
	EC2 : DeleteIpamExternalResourceVerificationToken
	EC2 : DeleteIpamPolicy
	EC2 : DeleteIpamPool
	EC2 : DeleteIpamPrefixListResolver
	EC2 : DeleteIpamPrefixListResolverTarget
	EC2 : DeleteIpamResourceDiscovery

Préfixe de service	Actions
	EC2 : DeleteIpamScope
	EC2 : DeleteKeyPair
	EC2 : DeleteLaunchTemplate
	EC2 : DeleteLaunchTemplateVersions
	EC2 : DeleteLocalGatewayRoute
	EC2 : DeleteLocalGatewayRouteTable
	EC2 : DeleteLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	EC2 : DeleteLocalGatewayRouteTableVpcAssociation
	EC2 : DeleteLocalGatewayVirtualInterface
	EC2 : DeleteLocalGatewayVirtualInterfaceGroup
	EC2 : DeleteManagedPrefixList
	EC2 : DeleteNatGateway
	EC2 : DeleteNetworkAcl
	EC2 : DeleteNetworkAclEntry
	EC2 : DeleteNetworkInsightsAccessScope
	EC2 : DeleteNetworkInsightsAccessScopeAnalysis
	EC2 : DeleteNetworkInsightsAnalysis
	EC2 : DeleteNetworkInsightsPath
	EC2 : DeleteNetworkInterface
	EC2 : DeleteNetworkInterfacePermission

Préfixe de service	Actions
	EC2 : DeletePlacementGroup
	EC2 : DeletePublicIpv4Pool
	EC2 : DeleteQueuedReservedInstances
	EC2 : DeleteRoute
	EC2 : DeleteRouteServer
	EC2 : DeleteRouteServerEndpoint
	EC2 : DeleteRouteServerPeer
	EC2 : DeleteRouteTable
	EC2 : DeleteSecondaryNetwork
	EC2 : DeleteSecondarySubnet
	EC2 : DeleteSecurityGroup
	EC2 : DeleteSpotDatafeedSubscription
	EC2 : DeleteSubnet
	EC2 : DeleteSubnetCidrReservation
	EC2 : DeleteTrafficMirrorFilter
	EC2 : DeleteTrafficMirrorFilterRule
	EC2 : DeleteTrafficMirrorSession
	EC2 : DeleteTrafficMirrorTarget
	EC2 : DeleteTransitGateway
	EC2 : DeleteTransitGatewayClientVpnAttachment
	EC2 : DeleteTransitGatewayConnect

Préfixe de service	Actions
	EC2 : DeleteTransitGatewayConnectPeer
	EC2 : DeleteTransitGatewayMeteringPolicy
	EC2 : DeleteTransitGatewayMeteringPolicyEntry
	EC2 : DeleteTransitGatewayMulticastDomain
	EC2 : DeleteTransitGatewayPeeringAttachment
	EC2 : DeleteTransitGatewayPolicyTable
	EC2 : DeleteTransitGatewayPrefixListReference
	EC2 : DeleteTransitGatewayRoute
	EC2 : DeleteTransitGatewayRouteTable
	EC2 : DeleteTransitGatewayRouteTableAnnouncement
	EC2 : DeleteTransitGatewayVpcAttachment
	EC2 : DeleteVerifiedAccessEndpoint
	EC2 : DeleteVerifiedAccessGroup
	EC2 : DeleteVerifiedAccessInstance
	EC2 : DeleteVerifiedAccessTrustProvider
	EC2 : DeleteVolume
	EC2 : DeleteVpc
	EC2 : DeleteVpcBlockPublicAccessExclusion
	EC2 : DeleteVpcEncryptionControl
	EC2 : DeleteVpcEndpointConnectionNotifications
	EC2 : DeleteVpcEndpointServiceConfigurations

Préfixe de service	Actions
	EC2 : DeleteVpcEndpoints
	EC2 : DeleteVpcPeeringConnection
	EC2 : DeleteVpnConcentrator
	EC2 : DeleteVpnConnection
	EC2 : DeleteVpnConnectionRoute
	EC2 : DeleteVpnGateway
	EC2 : DeprovisionByoipCidr
	EC2 : DeprovisionIpamByoasn
	EC2 : DeprovisionIpamPoolCidr
	EC2 : DeprovisionPublicIpv4PoolCidr
	EC2 : DeregisterImage
	EC2 : DeregisterInstanceEventNotificationAttributes
	EC2 : DeregisterTransitGatewayMulticastGroupMembers
	EC2 : DeregisterTransitGatewayMulticastGroupSources
	EC2 : DescribeAccountAttributes
	EC2 : DescribeAddressTransfers
	EC2 : DescribeAddresses
	EC2 : DescribeAddressesAttribute
	EC2 : DescribeAggregateIdFormat
	EC2 : DescribeAvailabilityZones
	EC2 : DescribeAwsNetworkPerformanceMetricSubscriptions

Préfixe de service	Actions
	EC2 : DescribeBundleTasks
	EC2 : DescribeByoipCidrs
	EC2 : DescribeCapacityBlockExtensionHistory
	EC2 : DescribeCapacityBlockExtensionOfferings
	EC2 : DescribeCapacityBlockStatus
	EC2 : DescribeCapacityBlocks
	EC2 : DescribeCapacityManagerDataExports
	EC2 : DescribeCapacityReservationBillingRequests
	EC2 : DescribeCapacityReservationCancellationQuotes
	EC2 : DescribeCapacityReservationFleets
	EC2 : DescribeCapacityReservationTopology
	EC2 : DescribeCapacityReservations
	EC2 : DescribeCarrierGateways
	EC2 : DescribeClassicLinkInstances
	EC2 : DescribeClientVpnAuthorizationRules
	EC2 : DescribeClientVpnConnections
	EC2 : DescribeClientVpnEndpoints
	EC2 : DescribeClientVpnRoutes
	EC2 : DescribeClientVpnTargetNetworks
	EC2 : DescribeCoipPools
	EC2 : DescribeConversionTasks

Préfixe de service	Actions
	EC2 : DescribeCustomerGateways
	EC2 : DescribeDeclarativePoliciesReports
	EC2 : DescribeDhcpOptions
	EC2 : DescribeEgressOnlyInternetGateways
	EC2 : DescribeElasticGpus
	EC2 : DescribeExportImageTasks
	EC2 : DescribeExportTasks
	EC2 : DescribeFastLaunchImages
	EC2 : DescribeFastSnapshotRestores
	EC2 : DescribeFleetHistory
	EC2 : DescribeFleetInstances
	EC2 : DescribeFleets
	EC2 : DescribeFlowLogs
	EC2 : DescribeFpgaImageAttribute
	EC2 : DescribeFpgaImages
	EC2 : DescribeHostReservationOfferings
	EC2 : DescribeHostReservations
	EC2 : DescribeHosts
	EC2 : DescribeIamInstanceProfileAssociations
	EC2 : DescribeIdFormat
	EC2 : DescribeIdentityIdFormat

Préfixe de service	Actions
	EC2 : DescribeImageAttribute
	EC2 : DescribeImageReferences
	EC2 : DescribeImageUsageReportEntries
	EC2 : DescribeImageUsageReports
	EC2 : DescribeImportImageTasks
	EC2 : DescribeImportSnapshotTasks
	EC2 : DescribeInstanceConnectEndpoints
	EC2 : DescribeInstanceCreditSpecifications
	EC2 : DescribeInstanceEventNotificationAttributes
	EC2 : DescribeInstanceEventWindows
	EC2 : DescribeInstanceImageMetadata
	EC2 : DescribeInstanceSqlHaHistoryStates
	EC2 : DescribeInstanceSqlHaStates
	EC2 : DescribeInstanceTopology
	EC2 : DescribeInstanceTypes
	EC2 : DescribeInternetGateways
	EC2 : DescribeIpamByoasn
	EC2 : DescribeIpamExternalResourceVerificationTokens
	EC2 : DescribeIpamPolicies
	EC2 : DescribeIpamPoolAllocations
	EC2 : DescribeIpamPools

Préfixe de service	Actions
	EC2 : DescribeIpamPrefixListResolverTargets
	EC2 : DescribeIpamPrefixListResolvers
	EC2 : DescribeIpamResourceDiscoveries
	EC2 : DescribeIpamResourceDiscoveryAssociations
	EC2 : DescribeIpamScopes
	EC2 : DescribeIpams
	EC2 : DescribeIpv6Pools
	EC2 : DescribeKeyPairs
	EC2 : DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations
	EC2 : DescribeLocalGatewayRouteTableVpcAssociations
	EC2 : DescribeLocalGatewayRouteTables
	EC2 : DescribeLocalGatewayVirtualInterfaceGroups
	EC2 : DescribeLocalGatewayVirtualInterfaces
	EC2 : DescribeLocalGateways
	EC2 : DescribeLockedSnapshots
	EC2 : DescribeMacHosts
	EC2 : DescribeMacModificationTasks
	EC2 : DescribeManagedPrefixLists
	EC2 : DescribeMovingAddresses
	EC2 : DescribeNatGateways

Préfixe de service	Actions
	EC2 : DescribeNetworkAcls
	EC2 : DescribeNetworkInsightsAccessScopeAnalyses
	EC2 : DescribeNetworkInsightsAccessScopes
	EC2 : DescribeNetworkInsightsAnalyses
	EC2 : DescribeNetworkInsightsPaths
	EC2 : DescribeNetworkInterfaceAttribute
	EC2 : DescribeNetworkInterfacePermissions
	EC2 : DescribeNetworkInterfaces
	EC2 : DescribeOutpostLags
	EC2 : DescribePlacementGroups
	EC2 : DescribePrefixLists
	EC2 : DescribePrincipalIdFormat
	EC2 : DescribePublicIpv4Pools
	EC2 : DescribeRegions
	EC2 : DescribeReplaceRootVolumeTasks
	EC2 : DescribeReservedInstances
	EC2 : DescribeReservedInstancesListings
	EC2 : DescribeReservedInstancesModifications
	EC2 : DescribeReservedInstancesOfferings
	EC2 : DescribeRouteServerEndpoints
	EC2 : DescribeRouteServerPeers

Préfixe de service	Actions
	EC2 : DescribeRouteServers
	EC2 : DescribeRouteTables
	EC2 : DescribeScheduledInstanceAvailability
	EC2 : DescribeScheduledInstances
	EC2 : DescribeSecondaryInterfaces
	EC2 : DescribeSecondaryNetworks
	EC2 : DescribeSecondarySubnets
	EC2 : DescribeSecurityGroupReferences
	EC2 : DescribeSecurityGroupRules
	EC2 : DescribeSecurityGroupVpcAssociations
	EC2 : DescribeSecurityGroups
	EC2 : DescribeServiceLinkVirtualInterfaces
	EC2 : DescribeSnapshotAttribute
	EC2 : DescribeSnapshotTierStatus
	EC2 : DescribeSpotDatafeedSubscription
	EC2 : DescribeSpotFleetInstances
	EC2 : DescribeSpotFleetRequestHistory
	EC2 : DescribeSpotFleetRequests
	EC2 : DescribeSpotInstanceRequests
	EC2 : DescribeSpotPriceHistory
	EC2 : DescribeStaleSecurityGroups

Préfixe de service	Actions
	EC2 : DescribeStoreImageTasks
	EC2 : DescribeTrafficMirrorFilterRules
	EC2 : DescribeTrafficMirrorFilters
	EC2 : DescribeTrafficMirrorSessions
	EC2 : DescribeTrafficMirrorTargets
	EC2 : DescribeTransitGatewayAttachments
	EC2 : DescribeTransitGatewayConnectPeers
	EC2 : DescribeTransitGatewayConnects
	EC2 : DescribeTransitGatewayMeteringPolicies
	EC2 : DescribeTransitGatewayMulticastDomains
	EC2 : DescribeTransitGatewayPeeringAttachments
	EC2 : DescribeTransitGatewayPolicyTables
	EC2 : DescribeTransitGatewayRouteTableAnnouncements
	EC2 : DescribeTransitGatewayRouteTables
	EC2 : DescribeTransitGatewayVpcAttachments
	EC2 : DescribeTransitGateways
	EC2 : DescribeTrunkInterfaceAssociations
	EC2 : DescribeVerifiedAccessEndpoints
	EC2 : DescribeVerifiedAccessGroups
	EC2 : DescribeVerifiedAccessInstanceLoggingConfigurations
	EC2 : DescribeVerifiedAccessInstances

Préfixe de service	Actions
	EC2 : DescribeVerifiedAccessTrustProviders
	EC2 : DescribeVolumeAttribute
	EC2 : DescribeVolumeStatus
	EC2 : DescribeVolumes
	EC2 : DescribeVolumesModifications
	EC2 : DescribeVpcAttribute
	EC2 : DescribeVpcBlockPublicAccessExclusions
	EC2 : DescribeVpcBlockPublicAccessOptions
	EC2 : DescribeVpcClassicLink
	EC2 : DescribeVpcClassicLinkDnsSupport
	EC2 : DescribeVpcEncryptionControls
	EC2 : DescribeVpcEndpointAssociations
	EC2 : DescribeVpcEndpointConnectionNotifications
	EC2 : DescribeVpcEndpointConnections
	EC2 : DescribeVpcEndpointServiceConfigurations
	EC2 : DescribeVpcEndpointServicePermissions
	EC2 : DescribeVpcEndpointServices
	EC2 : DescribeVpcEndpoints
	EC2 : DescribeVpcPeeringConnections
	EC2 : DescribeVpcs
	EC2 : DescribeVpnConcentrators

Préfixe de service	Actions
	EC2 : DescribeVpnConnections
	EC2 : DescribeVpnGateways
	EC2 : DetachClassicLinkVpc
	EC2 : DetachImageWatermark
	EC2 : DetachInternetGateway
	EC2 : DetachNetworkInterface
	EC2 : DetachVerifiedAccessTrustProvider
	EC2 : DetachVolume
	EC2 : DetachVpnGateway
	EC2 : DisableAddressTransfer
	EC2 : DisableAllowedImagesSettings
	EC2 : DisableAwsNetworkPerformanceMetricSubscription
	EC2 : DisableCapacityManager
	EC2 : DisableEbsEncryptionByDefault
	EC2 : DisableFastLaunch
	EC2 : DisableFastSnapshotRestores
	EC2 : DisableImage
	EC2 : DisableImageBlockPublicAccess
	EC2 : DisableImageDeprecation
	EC2 : DisableImageDeregistrationProtection
	EC2 : DisableInstanceSqlHaStandbyDetections

Préfixe de service	Actions
	EC2 : DisableIpamOrganizationAdminAccount
	EC2 : DisableIpamPolicy
	EC2 : DisableRouteServerPropagation
	EC2 : DisableSerialConsoleAccess
	EC2 : DisableSnapshotBlockPublicAccess
	EC2 : DisableTransitGatewayRouteTablePropagation
	EC2 : DisableVgwRoutePropagation
	EC2 : DisableVpcClassicLink
	EC2 : DisableVpcClassicLinkDnsSupport
	EC2 : DisassociateAddress
	EC2 : DisassociateCapacityReservationBillingOwner
	EC2 : DisassociateClientVpnTargetNetwork
	EC2 : DisassociateEnclaveCertificateIamRole
	EC2 : DisassociateIamInstanceProfile
	EC2 : DisassociateInstanceEventWindow
	EC2 : DisassociateIpamByoasn
	EC2 : DisassociateIpamResourceDiscovery
	EC2 : DisassociateNatGatewayAddress
	EC2 : DisassociateRouteServer
	EC2 : DisassociateRouteTable
	EC2 : DisassociateSecurityGroupVpc

Préfixe de service	Actions
	EC2 : DisassociateSubnetCidrBlock
	EC2 : DisassociateTransitGatewayMulticastDomain
	EC2 : DisassociateTransitGatewayPolicyTable
	EC2 : DisassociateTransitGatewayRouteTable
	EC2 : DisassociateTrunkInterface
	EC2 : DisassociateVpcCidrBlock
	EC2 : EnableAddressTransfer
	EC2 : EnableAllowedImagesSettings
	EC2 : EnableAwsNetworkPerformanceMetricSubscription
	EC2 : EnableCapacityManager
	EC2 : EnableEbsEncryptionByDefault
	EC2 : EnableFastLaunch
	EC2 : EnableFastSnapshotRestores
	EC2 : EnableImage
	EC2 : EnableImageBlockPublicAccess
	EC2 : EnableImageDeprecation
	EC2 : EnableImageDeregistrationProtection
	EC2 : EnableInstanceSqlHaStandbyDetections
	EC2 : EnableIpamOrganizationAdminAccount
	EC2 : EnableIpamPolicy
	EC2 : EnableReachabilityAnalyzerOrganizationSharing

Préfixe de service	Actions
	EC2 : EnableRouteServerPropagation
	EC2 : EnableSerialConsoleAccess
	EC2 : EnableSnapshotBlockPublicAccess
	EC2 : EnableTransitGatewayRouteTablePropagation
	EC2 : EnableVgwRoutePropagation
	ec2 : E/S EnableVolume
	EC2 : EnableVpcClassicLink
	EC2 : EnableVpcClassicLinkDnsSupport
	EC2 : ExportClientVpnClientCertificateRevocationList
	EC2 : ExportClientVpnClientConfiguration
	EC2 : ExportImage
	EC2 : ExportTransitGatewayRoutes
	EC2 : ExportVerifiedAccessInstanceClientConfiguration
	EC2 : GetActiveVpnTunnelStatus
	EC2 : GetAllowedImagesSettings
	EC2 : GetAssociatedEnclaveCertificateIamRoles
	EC2 : GetAssociatedIpv6PoolCidrs
	EC2 : GetAwsNetworkPerformanceData
	EC2 : GetCapacityManagerAttributes
	EC2 : GetCapacityManagerMetricData
	EC2 : GetCapacityManagerMetricDimensions

Préfixe de service	Actions
	EC2 : GetCapacityReservationUsage
	EC2 : GetCoipPoolUsage
	EC2 : GetConsoleOutput
	EC2 : GetConsoleScreenshot
	EC2 : GetDeclarativePoliciesReportSummary
	EC2 : GetDefaultCreditSpecification
	EC2 : GetEbsDefaultKmsKeyId
	EC2 : GetEbsEncryptionByDefault
	EC2 : GetEnabledIpamPolicy
	EC2 : GetFlowLogsIntegrationTemplate
	EC2 : GetGroupsForCapacityReservation
	EC2 : GetHostReservationPurchasePreview
	EC2 : GetImageAncestry
	EC2 : GetImageBlockPublicAccessState
	EC2 : GetInstanceMetadataDefaults
	EC2 : GetInstanceTpmEkPub
	EC2 : GetInstanceTypesFromInstanceRequirements
	EC2 : GetInstanceUefiData
	EC2 : GetIpamAddressHistory
	EC2 : GetIpamDiscoveredAccounts
	EC2 : GetIpamDiscoveredPublicAddresses

Préfixe de service	Actions
	EC2 : GetIpamDiscoveredResourceCidrs
	EC2 : GetIpamPolicyAllocationRules
	EC2 : GetIpamPolicyOrganizationTargets
	EC2 : GetIpamPoolAllocations
	EC2 : GetIpamPoolCidrs
	EC2 : GetIpamPrefixListResolverRules
	EC2 : GetIpamPrefixListResolverVersionEntries
	EC2 : GetIpamPrefixListResolverVersions
	EC2 : GetIpamResourceCidrs
	EC2 : GetLaunchTemplateData
	EC2 : GetManagedPrefixListAssociations
	EC2 : GetManagedPrefixListEntries
	EC2 : GetManagedResourceVisibility
	EC2 : GetNetworkInsightsAccessScopeAnalysisFindings
	EC2 : GetNetworkInsightsAccessScopeContent
	EC2 : GetPasswordData
	EC2 : GetReservedInstancesExchangeQuote
	EC2 : GetRouteServerAssociations
	EC2 : GetRouteServerPropagations
	EC2 : GetRouteServerRoutingDatabase
	EC2 : GetSecurityGroupsForVpc

Préfixe de service	Actions
	EC2 : GetSerialConsoleAccessStatus
	EC2 : GetSnapshotBlockPublicAccessState
	EC2 : GetSpotPlacementScores
	EC2 : GetSubnetCidrReservations
	EC2 : GetTransitGatewayAttachmentPropagations
	EC2 : GetTransitGatewayMeteringPolicyEntries
	EC2 : GetTransitGatewayMulticastDomainAssociations
	EC2 : GetTransitGatewayPolicyTableAssociations
	EC2 : GetTransitGatewayPolicyTableEntries
	EC2 : GetTransitGatewayPrefixListReferences
	EC2 : GetTransitGatewayRouteTableAssociations
	EC2 : GetTransitGatewayRouteTablePropagations
	EC2 : GetVerifiedAccessEndpointPolicy
	EC2 : GetVerifiedAccessEndpointTargets
	EC2 : GetVerifiedAccessGroupPolicy
	EC2 : GetVpcResourcesBlockingEncryptionEnforcement
	EC2 : GetVpnConnectionDeviceSampleConfiguration
	EC2 : GetVpnConnectionDeviceTypes
	EC2 : GetVpnTunnelReplacementStatus
	EC2 : ImportClientVpnClientCertificateRevocationList
	EC2 : ImportImage

Préfixe de service	Actions
	EC2 : ImportInstance
	EC2 : ImportKeyPair
	EC2 : ImportSnapshot
	EC2 : ImportVolume
	EC2 : ListImagesInRecycleBin
	EC2 : ListSnapshotsInRecycleBin
	EC2 : ListVolumesInRecycleBin
	EC2 : LockSnapshot
	EC2 : ModifyAddressAttribute
	EC2 : ModifyAvailabilityZoneGroup
	EC2 : ModifyCapacityReservation
	EC2 : ModifyCapacityReservationFleet
	EC2 : ModifyClientVpnEndpoint
	EC2 : ModifyDefaultCreditSpecification
	EC2 : ModifyEbsDefaultKmsKeyId
	EC2 : ModifyFleet
	EC2 : ModifyFpgaImageAttribute
	EC2 : ModifyHosts
	EC2 : ModifyIdFormat
	EC2 : ModifyIdentityIdFormat
	EC2 : ModifyImageAttribute

Préfixe de service	Actions
	EC2 : ModifyInstanceAttribute
	EC2 : ModifyInstanceCapacityReservationAttributes
	EC2 : ModifyInstanceConnectEndpoint
	EC2 : ModifyInstanceCpuOptions
	EC2 : ModifyInstanceCreditSpecification
	EC2 : ModifyInstanceEventStartTime
	EC2 : ModifyInstanceEventWindow
	EC2 : ModifyInstanceMaintenanceOptions
	EC2 : ModifyInstanceMetadataDefaults
	EC2 : ModifyInstanceMetadataOptions
	EC2 : ModifyInstanceNetworkPerformanceOptions
	EC2 : ModifyInstancePlacement
	EC2 : ModifyIpam
	EC2 : ModifyIpamPolicyAllocationRules
	EC2 : ModifyIpamPool
	EC2 : ModifyIpamPoolAllocation
	EC2 : ModifyIpamPrefixListResolver
	EC2 : ModifyIpamPrefixListResolverTarget
	EC2 : ModifyIpamResourceCidr
	EC2 : ModifyIpamResourceDiscovery
	EC2 : ModifyIpamScope

Préfixe de service	Actions
	EC2 : ModifyLaunchTemplate
	EC2 : ModifyLocalGatewayRoute
	EC2 : ModifyManagedPrefixList
	EC2 : ModifyManagedResourceVisibility
	EC2 : ModifyNetworkInterfaceAttribute
	EC2 : ModifyPrivateDnsNameOptions
	EC2 : ModifyPublicIpDnsNameOptions
	EC2 : ModifyReservedInstances
	EC2 : ModifyRouteServer
	EC2 : ModifySecurityGroupRules
	EC2 : ModifySnapshotAttribute
	EC2 : ModifySnapshotTier
	EC2 : ModifySpotFleetRequest
	EC2 : ModifySubnetAttribute
	EC2 : ModifyTrafficMirrorFilterNetworkServices
	EC2 : ModifyTrafficMirrorFilterRule
	EC2 : ModifyTrafficMirrorSession
	EC2 : ModifyTransitGateway
	EC2 : ModifyTransitGatewayMeteringPolicy
	EC2 : ModifyTransitGatewayPrefixListReference
	EC2 : ModifyTransitGatewayVpcAttachment

Préfixe de service	Actions
	EC2 : ModifyVerifiedAccessEndpoint
	EC2 : ModifyVerifiedAccessEndpointPolicy
	EC2 : ModifyVerifiedAccessGroup
	EC2 : ModifyVerifiedAccessGroupPolicy
	EC2 : ModifyVerifiedAccessInstance
	EC2 : ModifyVerifiedAccessInstanceLoggingConfiguration
	EC2 : ModifyVerifiedAccessTrustProvider
	EC2 : ModifyVolume
	EC2 : ModifyVolumeAttribute
	EC2 : ModifyVpcAttribute
	EC2 : ModifyVpcBlockPublicAccessExclusion
	EC2 : ModifyVpcBlockPublicAccessOptions
	EC2 : ModifyVpcEncryptionControl
	EC2 : ModifyVpcEndpoint
	EC2 : ModifyVpcEndpointConnectionNotification
	EC2 : ModifyVpcEndpointServiceConfiguration
	EC2 : ModifyVpcEndpointServicePayerResponsibility
	EC2 : ModifyVpcEndpointServicePermissions
	EC2 : ModifyVpcPeeringConnectionOptions
	EC2 : ModifyVpcTenancy
	EC2 : ModifyVpnConnection

Préfixe de service	Actions
	EC2 : ModifyVpnConnectionOptions
	EC2 : ModifyVpnTunnelCertificate
	EC2 : ModifyVpnTunnelOptions
	EC2 : MonitorInstances
	EC2 : MoveAddressToVpc
	EC2 : MoveByoipCidrToIpam
	EC2 : MoveCapacityReservationInstances
	EC2 : ProvisionByoipCidr
	EC2 : ProvisionIpamByoasn
	EC2 : ProvisionIpamPoolCidr
	EC2 : ProvisionPublicIpv4PoolCidr
	EC2 : PurchaseCapacityBlockExtension
	EC2 : PurchaseHostReservation
	EC2 : PurchaseReservedInstancesOffering
	EC2 : PurchaseScheduledInstances
	EC2 : RebootInstances
	EC2 : RegisterImage
	EC2 : RegisterInstanceEventNotificationAttributes
	EC2 : RegisterTransitGatewayMulticastGroupMembers
	EC2 : RegisterTransitGatewayMulticastGroupSources
	EC2 : RejectCapacityReservationBillingOwnership

Préfixe de service	Actions
	EC2 : RejectTransitGatewayClientVpnAttachment
	EC2 : RejectTransitGatewayMulticastDomainAssociations
	EC2 : RejectTransitGatewayPeeringAttachment
	EC2 : RejectTransitGatewayVpcAttachment
	EC2 : RejectVpcEndpointConnections
	EC2 : RejectVpcPeeringConnection
	EC2 : ReleaseAddress
	EC2 : ReleaseHosts
	EC2 : ReleaseIpamPoolAllocation
	EC2 : ReplaceIamInstanceProfileAssociation
	EC2 : ReplaceImageCriteriaInAllowedImagesSettings
	EC2 : ReplaceNetworkAclAssociation
	EC2 : ReplaceNetworkAclEntry
	EC2 : ReplaceRoute
	EC2 : ReplaceRouteTableAssociation
	EC2 : ReplaceTransitGatewayRoute
	EC2 : ReplaceVpnTunnel
	EC2 : ReportInstanceStatus
	EC2 : RequestSpotFleet
	EC2 : RequestSpotInstances
	EC2 : ResetAddressAttribute

Préfixe de service	Actions
	EC2 : ResetEbsDefaultKmsKeyId
	EC2 : ResetFpgaImageAttribute
	EC2 : ResetImageAttribute
	EC2 : ResetInstanceAttribute
	EC2 : ResetNetworkInterfaceAttribute
	EC2 : ResetSnapshotAttribute
	EC2 : RestoreAddressToClassic
	EC2 : RestoreImageFromRecycleBin
	EC2 : RestoreManagedPrefixListVersion
	EC2 : RestoreSnapshotFromRecycleBin
	EC2 : RestoreSnapshotTier
	EC2 : RestoreVolumeFromRecycleBin
	EC2 : RevokeClientVpnIngress
	EC2 : RevokeSecurityGroupEgress
	EC2 : RevokeSecurityGroupIngress
	EC2 : RunInstances
	EC2 : RunScheduledInstances
	EC2 : SearchLocalGatewayRoutes
	EC2 : SearchTransitGatewayMulticastGroups
	EC2 : SearchTransitGatewayRoutes
	EC2 : SendDiagnosticInterrupt

Préfixe de service	Actions
	EC2 : StartDeclarativePoliciesReport
	EC2 : StartInstances
	EC2 : StartNetworkInsightsAccessScopeAnalysis
	EC2 : StartNetworkInsightsAnalysis
	EC2 : StartVpcEndpointServicePrivateDnsVerification
	EC2 : TerminateClientVpnConnections
	EC2 : UnassignIpv6Addresses
	EC2 : UnassignPrivateIpAddresses
	EC2 : UnassignPrivateNatGatewayAddress
	EC2 : UnlockSnapshot
	EC2 : UnmonitorInstances
	EC2 : UpdateCapacityManagerOrganizationsAccess
	EC2 : UpdateInterruptibleCapacityReservationAllocation
	EC2 : UpdateSecurityGroupRuleDescriptionsEgress
	EC2 : UpdateSecurityGroupRuleDescriptionsIngress
	EC2 : WithdrawByoipCidr

Préfixe de service	Actions
ecr	ecr : BatchCheckLayerAvailability ecr : BatchDeleteImage ecr : BatchGetImage ecr : BatchGetRepositoryScanningConfiguration ecr : CompleteLayerUpload ecr : CreatePullThroughCacheRule ecr : CreateRepositoryCreationTemplate ecr : DeleteLifecyclePolicy ecr : DeletePullThroughCacheRule ecr : DeleteRegistryPolicy ecr : DeleteRepository ecr : DeleteRepositoryCreationTemplate ecr : DeleteRepositoryPolicy ecr : DeleteSigningConfiguration ecr : DescribeImageReplicationStatus ecr : DescribeImageScanFindings ecr : DescribeImages ecr : DescribePullThroughCacheRules ecr : DescribeRegistry ecr : DescribeRepositories ecr : DescribeRepositoryCreationTemplates

Préfixe de service	Actions
	ecr : GetAccountSetting ecr : GetAuthorizationToken ecr : GetDownloadUrlForLayer ecr : GetLifecyclePolicy ecr : GetLifecyclePolicyPreview ecr : GetRegistryPolicy ecr : GetRegistryScanningConfiguration ecr : GetRepositoryPolicy ecr : GetSigningConfiguration ecr : InitiateLayerUpload ecr : ListImages ecr : ListPullTimeUpdateExclusions ecr : PutAccountSetting ecr : PutImage ecr : PutImageScanningConfiguration ecr : PutRegistryPolicy ecr : PutRegistryScanningConfiguration ecr : PutReplicationConfiguration ecr : StartImageScan ecr : StartLifecyclePolicyPreview ecr : UpdatePullThroughCacheRule

Préfixe de service	Actions
	ecr : UpdateRepositoryCreationTemplate ecr : UploadLayerPart ecr : ValidatePullThroughCacheRule

Préfixe de service	Actions
ecr-public	ecr-public : BatchCheckLayerAvailability
	ecr-public : BatchDeleteImage
	ecr-public : CompleteLayerUpload
	ecr-public : CreateRepository
	ecr-public : DeleteRepository
	ecr-public : DeleteRepositoryPolicy
	ecr-public : DescribeImages
	ecr-public : DescribeRegistries
	ecr-public : DescribeRepositories
	ecr-public : GetAuthorizationToken
	ecr-public : GetRegistryCatalogData
	ecr-public : GetRepositoryCatalogData
	ecr-public : GetRepositoryPolicy
	ecr-public : InitiateLayerUpload
	ecr-public : PutImage
	ecr-public : PutRegistryCatalogData
	ecr-public : PutRepositoryCatalogData
	ecr-public : SetRepositoryPolicy
	ecr-public : UploadLayerPart

Préfixe de service	Actions
ecs	ecs : ContinueServiceDeployment ecs : CreateCapacityProvider ecs : CreateCluster ecs : CreateDaemon ecs : CreateService ecs : CreateTaskSet ecs : DeleteAccountSetting ecs : DeleteAttributes ecs : DeleteCapacityProvider ecs : DeleteCluster ecs : DeleteDaemon ecs : DeleteDaemonTaskDefinition ecs : DeleteExpressGatewayService ecs : DeleteService ecs : DeleteTaskDefinitions ecs : DeleteTaskSet ecs : DeregisterContainerInstance ecs : DeregisterTaskDefinition ecs : DescribeCapacityProviders ecs : DescribeClusters ecs : DescribeContainerInstances

Préfixe de service	Actions
	ecs : DescribeDaemon ecs : DescribeDaemonDeployments ecs : DescribeDaemonRevisions ecs : DescribeDaemonTaskDefinition ecs : DescribeExpressGatewayService ecs : DescribeServiceDeployments ecs : DescribeServiceRevisions ecs : DescribeServices ecs : DescribeTaskDefinition ecs : DescribeTaskSets ecs : DescribeTasks ecs : DiscoverPollEndpoint ecs : ExecuteCommand ecs : GetTaskProtection ecs : ListAccountSettings ecs : ListAttributes ecs : ListClusters ecs : ListContainerInstances ecs : ListDaemonDeployments ecs : ListDaemonTaskDefinitions ecs : ListDaemons

Préfixe de service	Actions
	ecs : ListServiceDeployments ecs : ListServices ecs : ListServicesByNamespace ecs : ListTaskDefinitionFamilies ecs : ListTaskDefinitions ecs : ListTasks ecs : PutAccountSetting ecs : PutAccountSettingDefault ecs : PutAttributes ecs : PutClusterCapacityProviders ecs : RegisterContainerInstance ecs : RegisterDaemonTaskDefinition ecs : RunTask ecs : StartTask ecs : StopServiceDeployment ecs : StopTask ecs : SubmitAttachmentStateChanges ecs : SubmitContainerStateChange ecs : SubmitTaskStateChange ecs : UpdateCapacityProvider ecs : UpdateCluster

Préfixe de service	Actions
	ecs : UpdateClusterSettings ecs : UpdateContainerAgent ecs : UpdateContainerInstancesState ecs : UpdateDaemon ecs : UpdateExpressGatewayService ecs : UpdateService ecs : UpdateServicePrimaryTaskSet ecs : UpdateTaskProtection ecs : UpdateTaskSet

Préfixe de service	Actions
eks	Eks : AssociateAccessPolicy
	Eks : AssociateEncryptionConfig
	Eks : AssociateIdentityProviderConfig
	Eks : CancelUpdate
	Eks : CreateAccessEntry
	Eks : CreateAddon
	Eks : CreateCluster
	Eks : CreateEksAnywhereSubscription
	Eks : CreateFargateProfile
	Eks : CreateNodegroup
	Eks : DeleteAccessEntry
	Eks : DeleteAddon
	Eks : DeleteCapability
	Eks : DeleteCluster
	Eks : DeleteEksAnywhereSubscription
	Eks : DeleteFargateProfile
	Eks : DeleteNodegroup
	Eks : DeletePodIdentityAssociation
	Eks : DeregisterCluster
	Eks : DescribeAccessEntry
	Eks : DescribeAddon

Préfixe de service	Actions
	Eks : DescribeAddonConfiguration
	Eks : DescribeAddonVersions
	Eks : DescribeCapability
	Eks : DescribeCluster
	Eks : DescribeClusterVersions
	Eks : DescribeEksAnywhereSubscription
	Eks : DescribeFargateProfile
	Eks : DescribeIdentityProviderConfig
	Eks : DescribeInsight
	Eks : DescribeInsightsRefresh
	Eks : DescribeNodegroup
	Eks : DescribePodIdentityAssociation
	Eks : DescribeUpdate
	Eks : DisassociateAccessPolicy
	Eks : DisassociateIdentityProviderConfig
	Eks : ListAccessEntries
	Eks : ListAccessPolicies
	Eks : ListAddons
	Eks : ListAssociatedAccessPolicies
	Eks : ListCapabilities
	Eks : ListClusters

Préfixe de service	Actions
	Eks : ListEksAnywhereSubscriptions
	Eks : ListFargateProfiles
	Eks : ListIdentityProviderConfigs
	Eks : ListInsights
	Eks : ListNodegroups
	Eks : ListPodIdentityAssociations
	Eks : ListUpdates
	Eks : RegisterCluster
	Eks : StartInsightsRefresh
	Eks : UpdateAccessEntry
	Eks : UpdateAddon
	Eks : UpdateCapability
	Eks : UpdateClusterConfig
	Eks : UpdateClusterVersion
	Eks : UpdateEksAnywhereSubscription
	Eks : UpdateNodegroupConfig
	Eks : UpdateNodegroupVersion
	Eks : UpdatePodIdentityAssociation

Préfixe de service	Actions
elasticache	Cache élastique : AuthorizeCacheSecurityGroupIngress Cache élastique : BatchApplyUpdateAction Cache élastique : BatchStopUpdateAction Cache élastique : CompleteMigration Cache élastique : CopyServerlessCacheSnapshot Cache élastique : CopySnapshot Cache élastique : CreateCacheCluster Cache élastique : CreateCacheParameterGroup Cache élastique : CreateCacheSecurityGroup Cache élastique : CreateCacheSubnetGroup Cache élastique : CreateGlobalReplicationGroup Cache élastique : CreateReplicationGroup Cache élastique : CreateServerlessCache Cache élastique : CreateServerlessCacheSnapshot Cache élastique : CreateSnapshot Cache élastique : CreateUser Cache élastique : CreateUserGroup Cache élastique : DecreaseNodeGroupsInGlobalReplicationGroup Cache élastique : DecreaseReplicaCount Cache élastique : DeleteCacheCluster Cache élastique : DeleteCacheParameterGroup

Préfixe de service	Actions
	Cache élastique : DeleteCacheSecurityGroup Cache élastique : DeleteCacheSubnetGroup Cache élastique : DeleteGlobalReplicationGroup Cache élastique : DeleteReplicationGroup Cache élastique : DeleteServerlessCache Cache élastique : DeleteServerlessCacheSnapshot Cache élastique : DeleteSnapshot Cache élastique : DeleteUser Cache élastique : DeleteUserGroup Cache élastique : DescribeCacheClusters Cache élastique : DescribeCacheEngineVersions Cache élastique : DescribeCacheParameterGroups Cache élastique : DescribeCacheParameters Cache élastique : DescribeCacheSecurityGroups Cache élastique : DescribeCacheSubnetGroups Cache élastique : DescribeEngineDefaultParameters Cache élastique : DescribeEvents Cache élastique : DescribeGlobalReplicationGroups Cache élastique : DescribeReplicationGroups Cache élastique : DescribeReservedCacheNodes Cache élastique : DescribeReservedCacheNodesOfferings

Préfixe de service	Actions
	Cache élastique : DescribeServerlessCacheSnapshots Cache élastique : DescribeServerlessCaches Cache élastique : DescribeServiceUpdates Cache élastique : DescribeSnapshots Cache élastique : DescribeUpdateActions Cache élastique : DescribeUserGroups Cache élastique : DescribeUsers Cache élastique : DisassociateGlobalReplicationGroup Cache élastique : ExportServerlessCacheSnapshot Cache élastique : FailoverGlobalReplicationGroup Cache élastique : IncreaseNodeGroupsInGlobalReplicationGroup Cache élastique : IncreaseReplicaCount Cache élastique : ListAllowedNodeTypeModifications Cache élastique : ModifyCacheCluster Cache élastique : ModifyCacheParameterGroup Cache élastique : ModifyCacheSubnetGroup Cache élastique : ModifyGlobalReplicationGroup Cache élastique : ModifyReplicationGroup Cache élastique : ModifyReplicationGroupShardConfiguration Cache élastique : ModifyServerlessCache Cache élastique : ModifyUser

Préfixe de service	Actions
	Cache élastique : ModifyUserGroup Cache élastique : PurchaseReservedCacheNodesOffering Cache élastique : RebalanceSlotsInGlobalReplicationGroup Cache élastique : RebootCacheCluster Cache élastique : ResetCacheParameterGroup Cache élastique : RevokeCacheSecurityGroupIngress Cache élastique : StartMigration Cache élastique : TestFailover Cache élastique : TestMigration

Préfixe de service	Actions
elasticbeanstalk	tige de haricot élastique : AbortEnvironmentUpdate
	tige de haricot élastique : ApplyEnvironmentManagedAction
	tige de haricot élastique : AssociateEnvironmentOperationsRole
	elasticbeanstalk:CheckDNSAvailability
	tige de haricot élastique : ComposeEnvironments
	tige de haricot élastique : CreateApplication
	tige de haricot élastique : CreateApplicationVersion
	tige de haricot élastique : CreateConfigurationTemplate
	tige de haricot élastique : CreateEnvironment
	tige de haricot élastique : CreatePlatformVersion
	tige de haricot élastique : CreateStorageLocation
	tige de haricot élastique : DeleteApplication
	tige de haricot élastique : DeleteApplicationVersion
	tige de haricot élastique : DeleteConfigurationTemplate
	tige de haricot élastique : DeleteEnvironmentConfiguration
	tige de haricot élastique : DeletePlatformVersion
	tige de haricot élastique : DescribeAccountAttributes
	tige de haricot élastique : DescribeApplicationVersions
	tige de haricot élastique : DescribeApplications
	tige de haricot élastique : DescribeConfigurationOptions
	tige de haricot élastique : DescribeConfigurationSettings

Préfixe de service	Actions
	tige de haricot élastique : DescribeEnvironmentHealth
	tige de haricot élastique : DescribeEnvironmentManagedActionHistory
	tige de haricot élastique : DescribeEnvironmentManagedActions
	tige de haricot élastique : DescribeEnvironmentResources
	tige de haricot élastique : DescribeEnvironments
	tige de haricot élastique : DescribeEvents
	tige de haricot élastique : DescribeInstancesHealth
	tige de haricot élastique : DescribePlatformVersion
	tige de haricot élastique : DisassociateEnvironmentOperationsRole
	tige de haricot élastique : ListAvailableSolutionStacks
	tige de haricot élastique : ListPlatformBranches
	tige de haricot élastique : ListPlatformVersions
	tige de haricot élastique : RebuildEnvironment
	tige de haricot élastique : RequestEnvironmentInfo
	tige de haricot élastique : RestartAppServer
	tige de haricot élastique : RetrieveEnvironmentInfo
	tige de haricot élastique : SwapEnvironmentCNAMEs
	tige de haricot élastique : TerminateEnvironment
	tige de haricot élastique : UpdateApplication
	tige de haricot élastique : UpdateApplicationResourceLifecycle

Préfixe de service	Actions
	tige de haricot élastique : UpdateApplicationVersion tige de haricot élastique : UpdateConfigurationTemplate tige de haricot élastique : UpdateEnvironment tige de haricot élastique : ValidateConfigurationSettings

Préfixe de service	Actions
elasticfilesystem	système de fichiers élastique : CreateAccessPoint système de fichiers élastique : CreateFileSystem système de fichiers élastique : CreateMountTarget système de fichiers élastique : CreateReplicationConfiguration système de fichiers élastique : DeleteAccessPoint système de fichiers élastique : DeleteFileSystem système de fichiers élastique : DeleteFileSystemPolicy système de fichiers élastique : DeleteMountTarget système de fichiers élastique : DeleteReplicationConfiguration système de fichiers élastique : DescribeAccessPoints système de fichiers élastique : DescribeAccountPreferences système de fichiers élastique : DescribeBackupPolicy système de fichiers élastique : DescribeFileSystemPolicy système de fichiers élastique : DescribeFileSystems système de fichiers élastique : DescribeLifecycleConfiguration système de fichiers élastique : DescribeMountTargetSecurityGroups système de fichiers élastique : DescribeMountTargets système de fichiers élastique : DescribeReplicationConfigurations système de fichiers élastique : ModifyMountTargetSecurityGroups système de fichiers élastique : PutAccountPreferences système de fichiers élastique : PutBackupPolicy

Préfixe de service	Actions
	système de fichiers élastique : PutFileSystemPolicy système de fichiers élastique : PutLifecycleConfiguration système de fichiers élastique : UpdateFileSystem système de fichiers élastique : UpdateFileSystemProtection

Préfixe de service	Actions
elasticloadbalancing	équilibrage de charge élastique : AddListenerCertificates
	équilibrage de charge élastique : AddTrustStoreRevocations
	équilibrage de charge élastique : ApplySecurityGroupsToLoadBalancer
	équilibrage de charge élastique : AttachLoadBalancerToSubnets
	équilibrage de charge élastique : ConfigureHealthCheck
	équilibrage de charge élastique : CreateAppCookieStickinessPolicy
	équilibrage de charge élastique : CreateLBCookieStickinessPolicy
	équilibrage de charge élastique : CreateListener
	équilibrage de charge élastique : CreateLoadBalancer
	équilibrage de charge élastique : CreateLoadBalancerListeners
	équilibrage de charge élastique : CreateLoadBalancerPolicy
	équilibrage de charge élastique : CreateRule
	équilibrage de charge élastique : CreateTargetGroup
	équilibrage de charge élastique : CreateTrustStore
	équilibrage de charge élastique : DeleteListener
	équilibrage de charge élastique : DeleteLoadBalancer
	équilibrage de charge élastique : DeleteLoadBalancerListeners
	équilibrage de charge élastique : DeleteLoadBalancerPolicy
	équilibrage de charge élastique : DeleteRule
	équilibrage de charge élastique : DeleteSharedTrustStoreAssociation

Préfixe de service	Actions
	équilibrage de charge élastique : DeleteTargetGroup
	équilibrage de charge élastique : DeleteTrustStore
	équilibrage de charge élastique : DeregisterInstancesFromLoad Balancer
	équilibrage de charge élastique : DeregisterTargets
	équilibrage de charge élastique : DescribeAccountLimits
	équilibrage de charge élastique : DescribeCapacityReservation
	équilibrage de charge élastique : DescribeInstanceHealth
	équilibrage de charge élastique : DescribeListenerAttributes
	équilibrage de charge élastique : DescribeListenerCertificates
	équilibrage de charge élastique : DescribeListeners
	équilibrage de charge élastique : DescribeLoadBalancerAttributes
	équilibrage de charge élastique : DescribeLoadBalancerPolicies
	équilibrage de charge élastique : DescribeLoadBalancerPolicyT ypes
	équilibrage de charge élastique : DescribeLoadBalancers
	équilibrage de charge élastique : DescribeRules
	elasticloadbalancing:DescribeSSLPolicies
	équilibrage de charge élastique : DescribeTargetGroupAttributes
	équilibrage de charge élastique : DescribeTargetGroups
	équilibrage de charge élastique : DescribeTargetHealth
	équilibrage de charge élastique : DescribeTrustStoreAssociations

Préfixe de service	Actions
	équilibrage de charge élastique : DescribeTrustStoreRevocations
	équilibrage de charge élastique : DescribeTrustStores
	équilibrage de charge élastique : DetachLoadBalancerFromSubnets
	équilibrage de charge élastique : DisableAvailabilityZonesFor LoadBalancer
	équilibrage de charge élastique : EnableAvailabilityZonesForL oadBalancer
	équilibrage de charge élastique : GetResourcePolicy
	équilibrage de charge élastique : GetTrustStoreCaCertificates Bundle
	équilibrage de charge élastique : GetTrustStoreRevocationContent
	équilibrage de charge élastique : ModifyCapacityReservation
	équilibrage de charge élastique : ModifyIpPools
	équilibrage de charge élastique : ModifyListener
	équilibrage de charge élastique : ModifyLoadBalancerAttributes
	équilibrage de charge élastique : ModifyRule
	équilibrage de charge élastique : ModifyTargetGroup
	équilibrage de charge élastique : ModifyTargetGroupAttributes
	équilibrage de charge élastique : ModifyTrustStore
	équilibrage de charge élastique : RegisterInstancesWithLoadBa lancer
	équilibrage de charge élastique : RegisterTargets
	équilibrage de charge élastique : RemoveListenerCertificates

Préfixe de service	Actions
	équilibrage de charge élastique : RemoveTrustStoreRevocations
	équilibrage de charge élastique : SetIpAddressType
	équilibrage de charge élastique : SetLoadBalancerListenerSSLCertificate
	équilibrage de charge élastique : SetLoadBalancerPoliciesForBackendServer
	équilibrage de charge élastique : SetLoadBalancerPoliciesOfListener
	équilibrage de charge élastique : SetRulePriorities
	équilibrage de charge élastique : SetSecurityGroups
	équilibrage de charge élastique : SetSubnets

Préfixe de service	Actions
emr-containers	conteneurs EMR : CancelJobRun conteneurs EMR : CreateJobTemplate conteneurs EMR : CreateManagedEndpoint conteneurs EMR : CreateSecurityConfiguration conteneurs EMR : CreateVirtualCluster conteneurs EMR : DeleteJobTemplate conteneurs EMR : DeleteManagedEndpoint conteneurs EMR : DeleteSecurityConfiguration conteneurs EMR : DeleteVirtualCluster conteneurs EMR : DescribeJobRun conteneurs EMR : DescribeJobTemplate conteneurs EMR : DescribeManagedEndpoint conteneurs EMR : DescribeSecurityConfiguration conteneurs EMR : DescribeVirtualCluster conteneurs EMR : GetManagedEndpointSessionCredentials conteneurs EMR : ListJobRuns Conteneurs EMR : ListJobTemplates Conteneurs EMR : ListManagedEndpoints Conteneurs EMR : ListSecurityConfigurations Conteneurs EMR : ListVirtualClusters Conteneurs EMR : StartJobRun

Préfixe de service	Actions
emr-serverless	emr sans serveur : CancelJobRun
	emr sans serveur : CreateApplication
	emr sans serveur : DeleteApplication
	emr sans serveur : GetApplication
	emr sans serveur : GetDashboardForJobRun
	emr sans serveur : GetJobRun
	emr sans serveur : GetResourceDashboard
	emr sans serveur : GetSession
	emr sans serveur : GetSessionEndpoint
	emr sans serveur : ListApplications
	emr sans serveur : ListJobRunAttempts
	emr sans serveur : ListJobRuns
	emr sans serveur : ListSessions
	emr sans serveur : StartApplication
	emr sans serveur : StartJobRun
	emr sans serveur : StopApplication
	emr sans serveur : TerminateSession
	emr sans serveur : UpdateApplication

Préfixe de service	Actions
es	Oui : AcceptInboundConnection Oui : AcceptInboundCrossClusterSearchConnection Oui : AssociatePackage Oui : AuthorizeVpcEndpointAccess Oui : CancelElasticsearchServiceSoftwareUpdate Oui : CancelServiceSoftwareUpdate Oui : CreateDomain Oui : CreateElasticsearchDomain Oui : CreateIndex Oui : CreateOutboundConnection Oui : CreateOutboundCrossClusterSearchConnection Oui : CreatePackage Oui : CreateVpcEndpoint Oui : DeleteDomain Oui : DeleteElasticsearchDomain Oui : DeleteElasticsearchServiceRole Oui : DeleteInboundConnection Oui : DeleteInboundCrossClusterSearchConnection Oui : DeleteIndex Oui : DeleteOutboundConnection Oui : DeleteOutboundCrossClusterSearchConnection

Préfixe de service	Actions
	Oui : DeletePackage Oui : DeleteVpcEndpoint Oui : DescribeDomain Oui : DescribeDomainAutoTunes Oui : DescribeDomainChangeProgress Oui : DescribeDomainConfig Oui : DescribeDomainHealth Oui : DescribeDomainNodes Oui : DescribeDomains Oui : DescribeDryRunProgress Oui : DescribeElasticsearchDomain Oui : DescribeElasticsearchDomainConfig Oui : DescribeElasticsearchDomains Oui : DescribeElasticsearchInstanceTypeLimits Oui : DescribeInboundConnections Oui : DescribeInboundCrossClusterSearchConnections Oui : DescribeInstanceTypeLimits Oui : DescribeOutboundConnections Oui : DescribeOutboundCrossClusterSearchConnections Oui : DescribePackages Oui : DescribeReservedElasticsearchInstanceOfferings

Préfixe de service	Actions
	Oui : DescribeReservedElasticsearchInstances Oui : DescribeReservedInstanceOfferings Oui : DescribeReservedInstances Oui : DescribeVpcEndpoints Oui : DissociatePackage Oui : DissociatePackages Oui : GetCompatibleElasticsearchVersions Oui : GetCompatibleVersions Oui : GetDataSource Oui : GetDomainMaintenanceStatus Oui : GetPackageVersionHistory Oui : GetUpgradeHistory Oui : GetUpgradeStatus Oui : ListDataSources Oui : ListDomainNames Oui : ListDomainsForPackage Oui : ListElasticsearchInstanceTypes Oui : ListElasticsearchVersions Oui : ListInsights Oui : ListInstanceTypeDetails Oui : ListPackagesForDomain

Préfixe de service	Actions
	Oui : ListScheduledActions Oui : ListVersions Oui : ListVpcEndpointAccess Oui : ListVpcEndpoints Oui : ListVpcEndpointsForDomain Oui : PurchaseReservedElasticsearchInstanceOffering Oui : PurchaseReservedInstanceOffering Oui : RejectInboundConnection Oui : RejectInboundCrossClusterSearchConnection Oui : RevokeVpcEndpointAccess Oui : RollbackServiceSoftwareUpdate Oui : StartDomainMaintenance Oui : StartElasticsearchServiceSoftwareUpdate Oui : StartServiceSoftwareUpdate Oui : UpdateDataSource Oui : UpdateDomainConfig Oui : UpdateElasticsearchDomainConfig Oui : UpdateIndex Oui : UpdatePackage Oui : UpdatePackageScope Oui : UpdateScheduledAction

Préfixe de service	Actions
	Oui : UpdateVpcEndpoint Oui : UpgradeDomain Oui : UpgradeElasticsearchDomain

Préfixe de service	Actions
events	événements : ActivateEventSource
	événements : CancelReplay
	événements : CreateApiDestination
	événements : CreateArchive
	événements : CreateConnection
	événements : CreateEndpoint
	événements : CreateEventBus
	événements : CreatePartnerEventSource
	événements : DeactivateEventSource
	événements : DeauthorizeConnection
	événements : DeleteApiDestination
	événements : DeleteArchive
	événements : DeleteConnection
	événements : DeleteEndpoint
	événements : DeleteEventBus
	événements : DeletePartnerEventSource
	événements : DeleteRule
	événements : DescribeApiDestination
	événements : DescribeArchive
	événements : DescribeConnection
	événements : DescribeEndpoint

Préfixe de service	Actions
	événements : DescribeEventBus
	événements : DescribeEventSource
	événements : DescribePartnerEventSource
	événements : DescribeReplay
	événements : DescribeRule
	événements : DisableRule
	événements : EnableRule
	événements : ListApiDestinations
	événements : ListArchives
	événements : ListConnections
	événements : ListEndpoints
	événements : ListEventBuses
	événements : ListEventSources
	événements : ListPartnerEventSourceAccounts
	événements : ListPartnerEventSources
	événements : ListReplays
	événements : ListRuleNamesByTarget
	événements : ListRules
	événements : ListTargetsByRule
	événements : PutPermission
	événements : PutRule

Préfixe de service	Actions
	événements : PutTargets
	événements : RemovePermission
	événements : RemoveTargets
	événements : StartReplay
	événements : TestEventPattern
	événements : UpdateApiDestination
	événements : UpdateArchive
	événements : UpdateConnection
	événements : UpdateEndpoint
	événements : UpdateEventBus

Préfixe de service	Actions
finspace	espace fin : CreateEnvironment espace fin : CreateKxChangeset espace fin : CreateKxCluster espace fin : CreateKxDatabase espace fin : CreateKxDataview espace fin : CreateKxEnvironment espace fin : CreateKxScalingGroup espace fin : CreateKxUser espace fin : CreateKxVolume espace fin : CreateUser espace fin : DeleteEnvironment espace fin : DeleteKxCluster espace fin : DeleteKxClusterNode espace fin : DeleteKxDatabase espace fin : DeleteKxDataview espace fin : DeleteKxEnvironment espace fin : DeleteKxScalingGroup espace fin : DeleteKxUser espace fin : DeleteKxVolume espace fin : GetEnvironment espace fin : GetKxChangeset

Préfixe de service	Actions
	espace fin : GetKxCluster espace fin : GetKxConnectionString espace fin : GetKxDatabase espace fin : GetKxDataview espace fin : GetKxEnvironment espace fin : GetKxScalingGroup espace fin : GetKxUser espace fin : GetKxVolume espace fin : GetLoadSampleDataSetGroupIntoEnvironmentStatus espace fin : GetUser espace fin : ListEnvironments espace fin : ListKxChangesets espace fin : ListKxClusterNodes espace fin : ListKxClusters espace fin : ListKxDatabases espace fin : ListKxDataviews espace fin : ListKxEnvironments espace fin : ListKxScalingGroups espace fin : ListKxUsers espace fin : ListKxVolumes espace fin : ListUsers

Préfixe de service	Actions
	espace fin : LoadSampleDataSetGroupIntoEnvironment espace fin : ResetUserPassword espace fin : UpdateEnvironment espace fin : UpdateKxClusterCodeConfiguration espace fin : UpdateKxClusterDatabases espace fin : UpdateKxDatabase espace fin : UpdateKxDataview espace fin : UpdateKxEnvironment espace fin : UpdateKxEnvironmentNetwork espace fin : UpdateKxUser espace fin : UpdateKxVolume espace fin : UpdateUser
firehose	lance à incendie : CreateDeliveryStream lance à incendie : DeleteDeliveryStream lance à incendie : DescribeDeliveryStream lance à incendie : ListDeliveryStreams lance à incendie : StartDeliveryStreamEncryption lance à incendie : StopDeliveryStreamEncryption lance à incendie : UpdateDestination

Préfixe de service	Actions
fis	poisson : CreateExperimentTemplate poisson : CreateTargetAccountConfiguration poisson : DeleteExperimentTemplate poisson : DeleteTargetAccountConfiguration poisson : GetAction poisson : GetExperiment poisson : GetExperimentTargetAccountConfiguration poisson : GetExperimentTemplate poisson : GetSafetyLever poisson : GetTargetAccountConfiguration poisson : GetTargetResourceType poisson : ListActions poisson : ListExperimentResolvedTargets poisson : ListExperimentTargetAccountConfigurations poisson : ListExperimentTemplates poisson : ListExperiments poisson : ListTargetAccountConfigurations poisson : ListTargetResourceTypes poisson : StartExperiment poisson : StopExperiment poisson : UpdateExperimentTemplate

Préfixe de service	Actions
	poisson : UpdateSafetyLeverState poisson : UpdateTargetAccountConfiguration

Préfixe de service	Actions
fms	fms : AssociateAdminAccount
	fms : AssociateThirdPartyFirewall
	fms : BatchAssociateResource
	fms : BatchDisassociateResource
	fms : DeleteAppsList
	fms : DeleteNotificationChannel
	fms : DeletePolicy
	fms : DeleteProtocolsList
	fms : DeleteResourceSet
	fms : DisassociateAdminAccount
	fms : DisassociateThirdPartyFirewall
	fms : GetAdminAccount
	fms : GetAdminScope
	fms : GetAppsList
	fms : GetComplianceDetail
	fms : GetNotificationChannel
	fms : GetPolicy
	fms : GetProtectionStatus
	fms : GetProtocolsList
	fms : GetResourceSet
	fms : GetThirdPartyFirewallAssociationStatus

Préfixe de service	Actions
	fms : GetViolationDetails
	fms : ListAdminAccountsForOrganization
	fms : ListAdminsManagingAccount
	fms : ListAppsLists
	fms : ListComplianceStatus
	fms : ListDiscoveredResources
	fms : ListMemberAccounts
	fms : ListPolicies
	fms : ListProtocolsLists
	fms : ListResourceSetResources
	fms : ListResourceSets
	fms : ListThirdPartyFirewallFirewallPolicies
	fms : PutAdminAccount
	fms : PutAppsList
	fms : PutNotificationChannel
	fms : PutPolicy
	fms : PutProtocolsList
	fms : PutResourceSet

Préfixe de service	Actions
frauddetector	détecteur de fraude : BatchCreateVariable détecteur de fraude : BatchGetVariable détecteur de fraude : CancelBatchImportJob détecteur de fraude : CancelBatchPredictionJob détecteur de fraude : CreateBatchImportJob détecteur de fraude : CreateBatchPredictionJob détecteur de fraude : CreateDetectorVersion détecteur de fraude : CreateList détecteur de fraude : CreateModel détecteur de fraude : CreateModelVersion détecteur de fraude : CreateRule détecteur de fraude : CreateVariable détecteur de fraude : DeleteBatchImportJob détecteur de fraude : DeleteBatchPredictionJob détecteur de fraude : DeleteDetector détecteur de fraude : DeleteDetectorVersion détecteur de fraude : DeleteEntityType détecteur de fraude : DeleteEvent détecteur de fraude : DeleteEventType détecteur de fraude : DeleteEventsByEventType détecteur de fraude : DeleteExternalModel

Préfixe de service	Actions
	détecteur de fraude : DeleteLabel détecteur de fraude : DeleteList détecteur de fraude : DeleteModel détecteur de fraude : DeleteModelVersion détecteur de fraude : DeleteOutcome détecteur de fraude : DeleteRule détecteur de fraude : DeleteVariable détecteur de fraude : DescribeDetector détecteur de fraude : DescribeModelVersions détecteur de fraude : GetBatchImportJobs détecteur de fraude : GetBatchPredictionJobs détecteur de fraude : GetDeleteEventsByEventTypeStatus détecteur de fraude : GetDetectorVersion détecteur de fraude : GetDetectors détecteur de fraude : GetEntityTypes détecteur de fraude : GetEvent détecteur de fraude : GetEventPrediction détecteur de fraude : GetEventPredictionMetadata détecteur de fraude : GetEventTypes détecteur de fraude : GetExternalModels détecteur de fraude : GetKMSEncryptionKey

Préfixe de service	Actions
	détecteur de fraude : GetLabels détecteur de fraude : GetListElements détecteur de fraude : GetListsMetadata détecteur de fraude : GetModelVersion détecteur de fraude : GetModels détecteur de fraude : GetOutcomes détecteur de fraude : GetRules détecteur de fraude : GetVariables détecteur de fraude : ListEventPredictions détecteur de fraude : PutDetector détecteur de fraude : PutEntityType détecteur de fraude : PutEventType détecteur de fraude : PutExternalModel détecteur de fraude : PutKMSEncryptionKey détecteur de fraude : PutLabel détecteur de fraude : PutOutcome détecteur de fraude : SendEvent détecteur de fraude : UpdateDetectorVersion détecteur de fraude : UpdateDetectorVersionMetadata détecteur de fraude : UpdateDetectorVersionStatus détecteur de fraude : UpdateEventLabel

Préfixe de service	Actions
	détecteur de fraude : UpdateList détecteur de fraude : UpdateModel détecteur de fraude : UpdateModelVersion détecteur de fraude : UpdateModelVersionStatus détecteur de fraude : UpdateRuleMetadata détecteur de fraude : UpdateRuleVersion détecteur de fraude : UpdateVariable

Préfixe de service	Actions
fsx	télécopie : AssociateFileSystemAliases
	télécopie : CancelDataRepositoryTask
	télécopie : CopyBackup
	télécopie : CreateDataRepositoryTask
	télécopie : CreateFileCache
	télécopie : CreateFileSystem
	télécopie : CreateFileSystemFromBackup
	télécopie : CreateSnapshot
	télécopie : CreateStorageVirtualMachine
	télécopie : CreateVolume
	télécopie : CreateVolumeFromBackup
	télécopie : DeleteBackup
	télécopie : DeleteFileCache
	télécopie : DeleteFileSystem
	télécopie : DeleteSnapshot
	télécopie : DeleteStorageVirtualMachine
	télécopie : DeleteVolume
	télécopie : DescribeBackups
	télécopie : DescribeDataRepositoryAssociations
	télécopie : DescribeDataRepositoryTasks
	télécopie : DescribeFileCaches

Préfixe de service	Actions
	télécopie : DescribeFileSystemAliases
	télécopie : DescribeFileSystems
	télécopie : DescribeS3AccessPointAttachments
	télécopie : DescribeSharedVpcConfiguration
	télécopie : DescribeSnapshots
	télécopie : DescribeStorageVirtualMachines
	télécopie : DescribeVolumes
	télécopie : DetachAndDeleteS3AccessPoint
	télécopie : DisassociateFileSystemAliases
	télécopie : ReleaseFileSystemNfsV3Locks
	télécopie : RestoreVolumeFromSnapshot
	télécopie : StartMisconfiguredStateRecovery
	télécopie : UpdateDataRepositoryAssociation
	télécopie : UpdateFileCache
	télécopie : UpdateFileSystem
	télécopie : UpdateSharedVpcConfiguration
	télécopie : UpdateSnapshot
	télécopie : UpdateStorageVirtualMachine
	télécopie : UpdateVolume

Préfixe de service	Actions
gamelift	gamelift : AcceptMatch gamelift : ClaimGameServer gamelift : CreateAlias gamelift : CreateBuild gamelift : CreateContainerGroupDefinition gamelift : CreateFleet gamelift : CreateFleetLocations gamelift : CreateGameServerGroup gamelift : CreateGameSession gamelift : CreateGameSessionQueue gamelift : CreateLocation gamelift : CreateMatchmakingConfiguration gamelift : CreateMatchmakingRuleSet gamelift : CreatePlayerSession gamelift : CreatePlayerSessions gamelift : CreateScript gamelift : CreateVpcPeeringAuthorization gamelift : CreateVpcPeeringConnection gamelift : DeleteAlias gamelift : DeleteBuild gamelift : DeleteContainerGroupDefinition

Préfixe de service	Actions
	gamelift : DeleteFleet gamelift : DeleteFleetLocations gamelift : DeleteGameServerGroup gamelift : DeleteGameSessionQueue gamelift : DeleteLocation gamelift : DeleteMatchmakingConfiguration gamelift : DeleteMatchmakingRuleSet gamelift : DeleteScalingPolicy gamelift : DeleteScript gamelift : DeleteVpcPeeringAuthorization gamelift : DeleteVpcPeeringConnection gamelift : DeregisterCompute gamelift : DeregisterGameServer gamelift : DescribeAlias gamelift : DescribeBuild gamelift : DescribeCompute gamelift : DescribeContainerFleet gamelift : DescribeContainerGroupDefinition gamelift : DescribeContainerGroupPortMappings gamelift : DescribeEC2InstanceLimits gamelift : DescribeFleetAttributes

Préfixe de service	Actions
	gamelift : DescribeFleetCapacity gamelift : DescribeFleetEvents gamelift : DescribeFleetLocationAttributes gamelift : DescribeFleetLocationCapacity gamelift : DescribeFleetLocationUtilization gamelift : DescribeFleetPortSettings gamelift : DescribeFleetUtilization gamelift : DescribeGameServer gamelift : DescribeGameServerGroup gamelift : DescribeGameServerInstances gamelift : DescribeGameSessionDetails gamelift : DescribeGameSessionPlacement gamelift : DescribeGameSessionQueues gamelift : DescribeGameSessions gamelift : DescribeInstances gamelift : DescribeMatchmaking gamelift : DescribeMatchmakingConfigurations gamelift : DescribeMatchmakingRuleSets gamelift : DescribePlayerSessions gamelift : DescribeRuntimeConfiguration gamelift : DescribeScalingPolicies

Préfixe de service	Actions
	gamelift : DescribeScript gamelift : DescribeVpcPeeringAuthorizations gamelift : DescribeVpcPeeringConnections gamelift : GetComputeAccess gamelift : GetComputeAuthToken gamelift : GetGameSessionLogUrl gamelift : GetInstanceAccess gamelift : GetPlayerConnectionDetails gamelift : ListAliases gamelift : ListBuilds gamelift : ListCompute gamelift : ListContainerFleets gamelift : ListContainerGroupDefinitionVersions gamelift : ListContainerGroupDefinitions gamelift : ListFleetDeployments gamelift : ListFleets gamelift : ListGameServerGroups gamelift : ListGameServers gamelift : ListLocations gamelift : ListScripts gamelift : PutScalingPolicy

Préfixe de service	Actions
	gamelift : RegisterCompute gamelift : RegisterGameServer gamelift : RequestUploadCredentials gamelift : ResolveAlias gamelift : ResumeGameServerGroup gamelift : SearchGameSessions gamelift : StartFleetActions gamelift : StartGameSessionPlacement gamelift : StartMatchBackfill gamelift : StartMatchmaking gamelift : StopFleetActions gamelift : StopGameSessionPlacement gamelift : StopMatchmaking gamelift : SuspendGameServerGroup gamelift : TerminateGameSession gamelift : UpdateAlias gamelift : UpdateBuild gamelift : UpdateContainerGroupDefinition gamelift : UpdateFleetAttributes gamelift : UpdateFleetCapacity gamelift : UpdateFleetPortSettings

Préfixe de service	Actions
	gamelift : UpdateGameServer gamelift : UpdateGameServerGroup gamelift : UpdateGameSession gamelift : UpdateGameSessionQueue gamelift : UpdateMatchmakingConfiguration gamelift : UpdateRuntimeConfiguration gamelift : UpdateScript gamelift : ValidateMatchmakingRuleSet

Préfixe de service	Actions
geo	géo : AssociateTrackerConsumer
	géo : BatchDeleteDevicePositionHistory
	géo : BatchDeleteGeofence
	géo : BatchEvaluateGeofences
	géo : BatchGetDevicePosition
	géo : BatchPutGeofence
	géo : BatchUpdateDevicePosition
	géo : CalculateRoute
	géo : CalculateRouteMatrix
	géo : CancelJob
	géo : CreateGeofenceCollection
	géo : CreateMap
	géo : CreatePlaceIndex
	géo : CreateRouteCalculator
	géo : CreateTracker
	géo : DeleteGeofenceCollection
	géo : DeleteKey
	géo : DeleteMap
	géo : DeletePlaceIndex
	géo : DeleteRouteCalculator
	géo : DeleteTracker

Préfixe de service	Actions
	géo : DescribeGeofenceCollection
	géo : DescribeKey
	géo : DescribeMap
	géo : DescribePlaceIndex
	géo : DescribeRouteCalculator
	géo : DescribeTracker
	géo : DisassociateTrackerConsumer
	géo : ForecastGeofenceEvents
	géo : GetDevicePosition
	géo : GetDevicePositionHistory
	géo : GetGeofence
	géo : GetJob
	géo : GetMapGlyphs
	géo : GetMapSprites
	géo : GetMapStyleDescriptor
	géo : GetMapTile
	géo : GetPlace
	géo : ListDevicePositions
	géo : ListGeofenceCollections
	géo : ListGeofences
	géo : ListJobs

Préfixe de service	Actions
	géo : ListKeys
	géo : ListMaps
	géo : ListPlaceIndexes
	géo : ListRouteCalculators
	géo : ListTrackerConsumers
	géo : ListTrackers
	géo : PutGeofence
	géo : SearchPlaceIndexForPosition
	géo : SearchPlaceIndexForSuggestions
	géo : SearchPlaceIndexForText
	géo : UpdateGeofenceCollection
	géo : UpdateKey
	géo : UpdateMap
	géo : UpdatePlaceIndex
	géo : UpdateRouteCalculator
	géo : UpdateTracker
	géo : VerifyDevicePosition

Préfixe de service	Actions
glacier	glacier : AbortMultipartUpload glacier : AbortVaultLock glacier : CompleteMultipartUpload glacier : CompleteVaultLock glacier : CreateVault glacier : DeleteArchive glacier : DeleteVault glacier : DeleteVaultAccessPolicy glacier : DeleteVaultNotifications glacier : DescribeJob glacier : DescribeVault glacier : GetDataRetrievalPolicy glacier : GetJobOutput glacier : GetVaultAccessPolicy glacier : GetVaultLock glacier : GetVaultNotifications glacier : InitiateJob glacier : InitiateMultipartUpload glacier : InitiateVaultLock glacier : ListJobs glacier : ListMultipartUploads

Préfixe de service	Actions
	glacier : ListParts glacier : ListProvisionedCapacity glacier : ListVaults glacier : PurchaseProvisionedCapacity glacier : SetDataRetrievalPolicy glacier : SetVaultAccessPolicy glacier : SetVaultNotifications glacier : UploadArchive glacier : UploadMultipartPart

Préfixe de service	Actions
grafana	Grafana : AssociateLicense
	Grafana : CreateWorkspace
	Grafana : CreateWorkspaceApiKey
	Grafana : CreateWorkspaceServiceAccount
	Grafana : CreateWorkspaceServiceAccountToken
	Grafana : DeleteWorkspace
	Grafana : DeleteWorkspaceApiKey
	Grafana : DeleteWorkspaceServiceAccount
	Grafana : DeleteWorkspaceServiceAccountToken
	Grafana : DescribeWorkspace
	Grafana : DescribeWorkspaceAuthentication
	Grafana : DescribeWorkspaceConfiguration
	Grafana : DisassociateLicense
	Grafana : ListPermissions
	Grafana : ListVersions
	Grafana : ListWorkspaceServiceAccountTokens
	Grafana : ListWorkspaceServiceAccounts
	Grafana : ListWorkspaces
	Grafana : UpdatePermissions
	Grafana : UpdateWorkspace
	Grafana : UpdateWorkspaceAuthentication

Préfixe de service	Actions
	Grafana : UpdateWorkspaceConfiguration

Préfixe de service	Actions
greengrass	herbe verte : AssociateRoleToGroup herbe verte : AssociateServiceRoleToAccount herbe verte : BatchAssociateClientDeviceWithCoreDevice herbe verte : BatchDisassociateClientDeviceFromCoreDevice herbe verte : CancelDeployment herbe verte : CreateComponentVersion herbe verte : CreateConnectorDefinition herbe verte : CreateConnectorDefinitionVersion herbe verte : CreateCoreDefinition herbe verte : CreateCoreDefinitionVersion herbe verte : CreateDeployment herbe verte : CreateDeviceDefinition herbe verte : CreateDeviceDefinitionVersion herbe verte : CreateFunctionDefinition herbe verte : CreateFunctionDefinitionVersion herbe verte : CreateGroup herbe verte : CreateGroupCertificateAuthority herbe verte : CreateGroupVersion herbe verte : CreateLoggerDefinition herbe verte : CreateLoggerDefinitionVersion herbe verte : CreateResourceDefinition

Préfixe de service	Actions
	herbe verte : CreateResourceDefinitionVersion herbe verte : CreateSoftwareUpdateJob herbe verte : CreateSubscriptionDefinition herbe verte : CreateSubscriptionDefinitionVersion herbe verte : DeleteComponent herbe verte : DeleteConnectorDefinition herbe verte : DeleteCoreDefinition herbe verte : DeleteCoreDevice herbe verte : DeleteDeployment herbe verte : DeleteDeviceDefinition herbe verte : DeleteFunctionDefinition herbe verte : DeleteGroup herbe verte : DeleteLoggerDefinition herbe verte : DeleteResourceDefinition herbe verte : DeleteSubscriptionDefinition herbe verte : DescribeComponent herbe verte : DisassociateRoleFromGroup herbe verte : DisassociateServiceRoleFromAccount herbe verte : GetAssociatedRole herbe verte : GetBulkDeploymentStatus herbe verte : GetComponent

Préfixe de service	Actions
	herbe verte : GetComponentVersionArtifact herbe verte : GetConnectivityInfo herbe verte : GetConnectorDefinition herbe verte : GetConnectorDefinitionVersion herbe verte : GetCoreDefinition herbe verte : GetCoreDefinitionVersion herbe verte : GetCoreDevice herbe verte : GetDeployment herbe verte : GetDeploymentStatus herbe verte : GetDeviceDefinition herbe verte : GetDeviceDefinitionVersion herbe verte : GetFunctionDefinition herbe verte : GetFunctionDefinitionVersion herbe verte : GetGroup herbe verte : GetGroupCertificateAuthority herbe verte : GetGroupCertificateConfiguration herbe verte : GetGroupVersion herbe verte : GetLoggerDefinition herbe verte : GetLoggerDefinitionVersion herbe verte : GetResourceDefinition herbe verte : GetResourceDefinitionVersion

Préfixe de service	Actions
	herbe verte : GetServiceRoleForAccount herbe verte : GetSubscriptionDefinition herbe verte : GetSubscriptionDefinitionVersion herbe verte : GetThingRuntimeConfiguration herbe verte : ListBulkDeploymentDetailedReports herbe verte : ListBulkDeployments herbe verte : ListClientDevicesAssociatedWithCoreDevice herbe verte : ListComponentVersions herbe verte : ListComponents herbe verte : ListConnectorDefinitionVersions herbe verte : ListConnectorDefinitions herbe verte : ListCoreDefinitionVersions herbe verte : ListCoreDefinitions herbe verte : ListCoreDevices herbe verte : ListDeployments herbe verte : ListDeviceDefinitionVersions herbe verte : ListDeviceDefinitions herbe verte : ListEffectiveDeployments herbe verte : ListFunctionDefinitionVersions herbe verte : ListFunctionDefinitions herbe verte : ListGroupCertificateAuthorities

Préfixe de service	Actions
	herbe verte : ListGroupVersions herbe verte : ListGroups herbe verte : ListInstalledComponents herbe verte : ListLoggerDefinitionVersions herbe verte : ListLoggerDefinitions herbe verte : ListResourceDefinitionVersions herbe verte : ListResourceDefinitions herbe verte : ListSubscriptionDefinitionVersions herbe verte : ListSubscriptionDefinitions herbe verte : ResetDeployments herbe verte : StartBulkDeployment herbe verte : StopBulkDeployment herbe verte : UpdateConnectivityInfo herbe verte : UpdateConnectorDefinition herbe verte : UpdateCoreDefinition herbe verte : UpdateDeviceDefinition herbe verte : UpdateFunctionDefinition herbe verte : UpdateGroup herbe verte : UpdateGroupCertificateConfiguration herbe verte : UpdateLoggerDefinition herbe verte : UpdateResourceDefinition

Préfixe de service	Actions
	herbe verte : UpdateSubscriptionDefinition herbe verte : UpdateThingRuntimeConfiguration

Préfixe de service	Actions
groundstation	station au sol : CancelContact
	station au sol : CreateConfig
	station au sol : CreateDataflowEndpointGroup
	station au sol : CreateDataflowEndpointGroupV2
	station au sol : CreateEphemeris
	station au sol : CreateMissionProfile
	station au sol : DeleteConfig
	station au sol : DeleteDataflowEndpointGroup
	station au sol : DeleteEphemeris
	station au sol : DeleteMissionProfile
	station au sol : DescribeContact
	station au sol : DescribeContactVersion
	station au sol : DescribeEphemeris
	station au sol : GetConfig
	station au sol : GetDataflowEndpointGroup
	station au sol : GetMinuteUsage
	station au sol : GetMissionProfile
	station au sol : GetSatellite
	station au sol : ListAntennas
	station au sol : ListConfigs
	station au sol : ListContacts

Préfixe de service	Actions
	station au sol : ListDataflowEndpointGroups
	station au sol : ListEphemerides
	station au sol : ListGroundStationReservations
	station au sol : ListGroundStations
	station au sol : ListMissionProfiles
	station au sol : ListSatellites
	station au sol : RegisterAgent
	station au sol : ReserveContact
	station au sol : UpdateAgentStatus
	station au sol : UpdateConfig
	station au sol : UpdateEphemeris
	station au sol : UpdateMissionProfile

Préfixe de service	Actions
guardduty	devoir de garde : AcceptAdministratorInvitation devoir de garde : AcceptInvitation devoir de garde : ArchiveFindings devoir de garde : CreateDetector devoir de garde : CreateFilter guardduty:CreateIPSet devoir de garde : CreateInvestigation devoir de garde : CreateMalwareProtectionPlan devoir de garde : CreateMembers devoir de garde : CreatePublishingDestination devoir de garde : CreateSampleFindings devoir de garde : CreateThreatEntitySet devoir de garde : CreateThreatIntelSet devoir de garde : CreateTrustedEntitySet devoir de garde : DeclineInvitations devoir de garde : DeleteDetector devoir de garde : DeleteFilter guardduty:DeleteIPSet devoir de garde : DeleteInvitations devoir de garde : DeleteMalwareProtectionPlan devoir de garde : DeleteMembers

Préfixe de service	Actions
	devoir de garde : DeletePublishingDestination devoir de garde : DeleteThreatEntitySet devoir de garde : DeleteThreatIntelSet devoir de garde : DeleteTrustedEntitySet devoir de garde : DescribeMalwareScans devoir de garde : DescribeOrganizationConfiguration devoir de garde : DescribePublishingDestination devoir de garde : DisableOrganizationAdminAccount devoir de garde : DisassociateFromAdministratorAccount devoir de garde : DisassociateFromMasterAccount devoir de garde : DisassociateMembers devoir de garde : EnableOrganizationAdminAccount devoir de garde : GetAdministratorAccount devoir de garde : GetCoverageStatistics devoir de garde : GetDetector devoir de garde : GetFilter devoir de garde : GetFindings devoir de garde : GetFindingsStatistics guardduty:GetIPSet devoir de garde : GetInvestigation devoir de garde : GetInvitationsCount

Préfixe de service	Actions
	devoir de garde : GetMalwareProtectionPlan devoir de garde : GetMalwareScan devoir de garde : GetMalwareScanSettings devoir de garde : GetMasterAccount devoir de garde : GetMemberDetectors devoir de garde : GetMembers devoir de garde : GetOrganizationStatistics devoir de garde : GetRemainingFreeTrialDays devoir de garde : GetThreatEntitySet devoir de garde : GetThreatIntelSet devoir de garde : GetTrustedEntitySet devoir de garde : GetUsageStatistics devoir de garde : InviteMembers devoir de garde : ListCoverage devoir de garde : ListDetectors devoir de garde : ListFilters devoir de garde : ListFindings guardduty:ListIPSets devoir de garde : ListInvestigations devoir de garde : ListInvitations devoir de garde : ListMalwareProtectionPlans

Préfixe de service	Actions
	devoir de garde : ListMalwareScans devoir de garde : ListMembers devoir de garde : ListOrganizationAdminAccounts devoir de garde : ListPublishingDestinations devoir de garde : ListThreatEntitySets devoir de garde : ListThreatIntelSets devoir de garde : ListTrustedEntitySets devoir de garde : StartMalwareScan devoir de garde : StartMonitoringMembers devoir de garde : StopMonitoringMembers devoir de garde : UnarchiveFindings devoir de garde : UpdateDetector devoir de garde : UpdateFilter devoir de garde : UpdateFindingsFeedback guardduty:UpdateIPSet devoir de garde : UpdateMalwareProtectionPlan devoir de garde : UpdateMalwareScanSettings devoir de garde : UpdateMemberDetectors devoir de garde : UpdateOrganizationConfiguration devoir de garde : UpdatePublishingDestination devoir de garde : UpdateThreatEntitySet

Préfixe de service	Actions
	devoir de garde : UpdateThreatIntelSet devoir de garde : UpdateTrustedEntitySet

Préfixe de service	Actions
healthlake	lac de santé : CancelFHIRExportJobWithDelete
	lac de santé : CreateDataTransformationProfile
	healthlake:CreateFHIRDatastore
	lac de santé : CreateResource
	lac de santé : DeleteDataTransformationProfile
	healthlake:DeleteFHIRDatastore
	lac de santé : DeleteResource
	lac de santé : DescribeDataTransformationJob
	healthlake:DescribeFHIRDatastore
	lac de santé : DescribeFHIRExportJob
	lac de santé : DescribeFHIRExportJobWithGet
	lac de santé : DescribeFHIRImportJob
	lac de santé : GetCapabilities
	lac de santé : GetDataTransformationProfile
	lac de santé : ListDataTransformationJobs
	lac de santé : ListDataTransformationProfileVersions
	lac de santé : ListDataTransformationProfiles
	healthlake:ListFHIRDatastores
	lac de santé : ListFHIRExportJobs
	lac de santé : ListFHIRImportJobs
	lac de santé : PublishDataTransformationProfile

Préfixe de service	Actions
	lac de santé : ReadResource
	lac de santé : SearchEverything
	lac de santé : SearchWithGet
	lac de santé : SearchWithPost
	lac de santé : StartFHIRExportJob
	lac de santé : StartFHIRExportJobWithPost
	lac de santé : StartFHIRImportJob
	lac de santé : UpdateDataTransformationProfile
	Healthlake : mise à jour de la banque de données HIR
	lac de santé : UpdateProfileWithAgent
	lac de santé : UpdateResource

Préfixe de service	Actions
honeycode	code d'identification : BatchCreateTableRows code d'identification : BatchDeleteTableRows code d'identification : BatchUpdateTableRows code d'identification : BatchUpsertTableRows code d'identification : DescribeTableDataImportJob code d'identification : GetScreenData code d'identification : InvokeScreenAutomation code d'identification : ListTableColumns code d'identification : ListTableRows code d'identification : ListTables code d'identification : QueryTableRows code d'identification : StartTableDataImportJob

Préfixe de service	Actions
iam	iam : AddClientIDToOpenIDConnectProvider
	iam : AddRoleToInstanceProfile
	iam : AddUserToGroup
	iam : AttachGroupPolicy
	iam : AttachRolePolicy
	iam : AttachUserPolicy
	iam : ChangePassword
	iam : CreateAccessKey
	iam : CreateAccountAlias
	iam : CreateGroup
	iam : CreateInstanceProfile
	iam : CreateLoginProfile
	iam : CreateOpenIDConnectProvider
	iam : CreatePolicy
	iam : CreatePolicyVersion
	iam : CreateRole
	iam:CreateSAMLProvider
	iam : CreateServiceLinkedRole
	iam : CreateServiceSpecificCredential
	iam : CreateUser
	iam : CreateVirtualMFADevice

Préfixe de service	Actions
	iam:DeactivateMFADevice
	iam : DeleteAccessKey
	iam : DeleteAccountAlias
	iam : DeleteAccountPasswordPolicy
	iam : DeleteCloudFrontPublicKey
	iam : DeleteGroup
	iam : DeleteGroupPolicy
	iam : DeleteInstanceProfile
	iam : DeleteLoginProfile
	iam : DeleteOpenIDConnectProvider
	iam : DeletePolicy
	iam : DeletePolicyVersion
	iam : DeleteRole
	iam : DeleteRolePermissionsBoundary
	iam : DeleteRolePolicy
	iam:DeleteSAMLProvider
	iam : DeleteSSHPublicKey
	iam : DeleteServerCertificate
	iam : DeleteServiceLinkedRole
	iam : DeleteServiceSpecificCredential
	iam : DeleteSigningCertificate

Préfixe de service	Actions
	iam : DeleteUser
	iam : DeleteUserPermissionsBoundary
	iam : DeleteUserPolicy
	iam : DeleteVirtualMFADevice
	iam : DetachGroupPolicy
	iam : DetachRolePolicy
	iam : DetachUserPolicy
	iam : DisableOrganizationsRootCredentialsManagement
	iam : DisableOrganizationsRootSessions
	iam : DisableOutboundWebIdentityFederation
	iam:EnableMFADevice
	iam : EnableOrganizationsRootCredentialsManagement
	iam : EnableOrganizationsRootSessions
	iam : EnableOutboundWebIdentityFederation
	iam : GenerateCredentialReport
	iam : GenerateOrganizationsAccessReport
	iam : GenerateServiceLastAccessedDetails
	iam : GetAccessKeyLastUsed
	iam : GetAccountAuthorizationDetails
	iam : GetAccountEmailAddress
	iam : GetAccountName

Préfixe de service	Actions
	iam : GetAccountPasswordPolicy
	iam : GetAccountSummary
	iam : GetCloudFrontPublicKey
	iam : GetContextKeysForCustomPolicy
	iam : GetContextKeysForPrincipalPolicy
	iam : GetCredentialReport
	iam : GetGroup
	iam : GetGroupPolicy
	iam : GetInstanceProfile
	iam : GetLoginProfile
	iam:GetMFADevice
	iam : GetOpenIDConnectProvider
	iam : GetOrganizationsAccessReport
	iam : GetOutboundWebIdentityFederationInfo
	iam : GetPolicy
	iam : GetPolicyVersion
	iam : GetRole
	iam : GetRolePolicy
	iam:GetSAMLProvider
	iam : GetSSHPublicKey
	iam : GetServerCertificate

Préfixe de service	Actions
	iam : GetServiceLastAccessedDetails
	iam : GetServiceLastAccessedDetailsWithEntities
	iam : GetServiceLinkedRoleDeletionStatus
	iam : GetUser
	iam : GetUserPolicy
	iam : ListAccessKeys
	iam : ListAccountAliases
	iam : ListAttachedGroupPolicies
	iam : ListAttachedRolePolicies
	iam : ListAttachedUserPolicies
	iam : ListCloudFrontPublicKeys
	iam : ListDelegationRequests
	iam : ListEntitiesForPolicy
	iam : ListGroupPolicies
	iam : ListGroups
	iam : ListGroupsForUser
	iam : ListInstanceProfiles
	iam : ListInstanceProfilesForRole
	iam:ListMFADevices
	iam : ListOpenIDConnectProviders
	iam : ListOrganizationsFeatures

Préfixe de service	Actions
	iam : ListPolicies
	iam : ListPoliciesGrantingServiceAccess
	iam : ListPolicyVersions
	iam : ListRolePolicies
	iam : ListRoles
	iam:ListSAMLProviders
	iam : ListSSHPublicKeys
	iam : ListSTSRegionalEndpointsStatus
	iam : ListServerCertificates
	iam : ListServiceSpecificCredentials
	iam : ListSigningCertificates
	iam : ListUserPolicies
	iam : ListUsers
	iam : ListVirtualMFADevices
	iam : PutGroupPolicy
	iam : PutRolePermissionsBoundary
	iam : PutRolePolicy
	iam : PutUserPermissionsBoundary
	iam : PutUserPolicy
	iam : RemoveClientIDFromOpenIDConnectProvider
	iam : RemoveRoleFromInstanceProfile

Préfixe de service	Actions
	iam : RemoveUserFromGroup
	iam : ResetServiceSpecificCredential
	iam:ResyncMFADevice
	iam : SetDefaultPolicyVersion
	iam : SetSTSRegionalEndpointStatus
	iam : SetSecurityTokenServicePreferences
	iam : SimulateCustomPolicy
	iam : SimulatePrincipalPolicy
	iam : UpdateAccessKey
	iam : UpdateAccountEmailAddress
	iam : UpdateAccountName
	iam : UpdateAccountPasswordPolicy
	iam : UpdateAssumeRolePolicy
	iam : UpdateCloudFrontPublicKey
	iam : UpdateGroup
	iam : UpdateLoginProfile
	iam : UpdateOpenIDConnectProviderThumbprint
	iam : UpdateRole
	iam : UpdateRoleDescription
	iam:UpdateSAMLProvider
	iam : UpdateSSHPublicKey

Préfixe de service	Actions
	iam : UpdateServerCertificate iam : UpdateServiceSpecificCredential iam : UpdateSigningCertificate iam : UpdateUser iam : UploadCloudFrontPublicKey iam : UploadSSHPublicKey iam : UploadServerCertificate iam : UploadSigningCertificate

Préfixe de service	Actions
identitystore	boutique d'identité : CreateGroup boutique d'identité : CreateGroupMembership boutique d'identité : CreateUser boutique d'identité : DeleteGroup boutique d'identité : DeleteGroupMembership boutique d'identité : DeleteUser boutique d'identité : DescribeGroup boutique d'identité : DescribeGroupMembership boutique d'identité : DescribeUser boutique d'identité : GetGroupId boutique d'identité : GetGroupMembershipId boutique d'identité : GetUserId boutique d'identité : IsMemberInGroups boutique d'identité : ListGroupMemberships boutique d'identité : ListGroupMembershipsForMember boutique d'identité : ListGroups boutique d'identité : ListUsers boutique d'identité : UpdateGroup boutique d'identité : UpdateUser

Préfixe de service	Actions
imagebuilder	générateur d'images : CancelImageCreation
	générateur d'images : CancelLifecycleExecution
	générateur d'images : CreateComponent
	générateur d'images : CreateContainerRecipe
	générateur d'images : CreateDistributionConfiguration
	générateur d'images : CreateImage
	générateur d'images : CreateImagePipeline
	générateur d'images : CreateImageRecipe
	générateur d'images : CreateInfrastructureConfiguration
	générateur d'images : CreateLifecyclePolicy
	générateur d'images : CreateWorkflow
	générateur d'images : DeleteComponent
	générateur d'images : DeleteContainerRecipe
	générateur d'images : DeleteDistributionConfiguration
	générateur d'images : DeleteImage
	générateur d'images : DeleteImagePipeline
	générateur d'images : DeleteImageRecipe
	générateur d'images : DeleteInfrastructureConfiguration
	générateur d'images : DeleteLifecyclePolicy
	générateur d'images : DeleteWorkflow
	générateur d'images : DistributeImage

Préfixe de service	Actions
	générateur d'images : GetComponentPolicy
	générateur d'images : GetContainerRecipePolicy
	générateur d'images : GetImagePolicy
	générateur d'images : GetImageRecipePolicy
	générateur d'images : GetLifecycleExecution
	générateur d'images : GetLifecyclePolicy
	générateur d'images : GetMarketplaceResource
	générateur d'images : GetWorkflowExecution
	générateur d'images : GetWorkflowStepExecution
	générateur d'images : ImportComponent
	générateur d'images : ImportDiskImage
	générateur d'images : ImportVmImage
	générateur d'images : ListComponentBuildVersions
	générateur d'images : ListComponents
	générateur d'images : ListContainerRecipes
	générateur d'images : ListDistributionConfigurations
	générateur d'images : ListImageBuildVersions
	générateur d'images : ListImagePackages
	générateur d'images : ListImagePipelineImages
	générateur d'images : ListImagePipelines
	générateur d'images : ListImageRecipes

Préfixe de service	Actions
	générateur d'images : ListImageScanFindingAggregations
	générateur d'images : ListImageScanFindings
	générateur d'images : ListImages
	générateur d'images : ListInfrastructureConfigurations
	générateur d'images : ListLifecycleExecutionResources
	générateur d'images : ListLifecycleExecutions
	générateur d'images : ListLifecyclePolicies
	générateur d'images : ListWaitingWorkflowSteps
	générateur d'images : ListWorkflowExecutions
	générateur d'images : ListWorkflowStepExecutions
	générateur d'images : ListWorkflows
	générateur d'images : PutComponentPolicy
	générateur d'images : PutContainerRecipePolicy
	générateur d'images : PutImagePolicy
	générateur d'images : PutImageRecipePolicy
	générateur d'images : RetryImage
	générateur d'images : SendWorkflowStepAction
	générateur d'images : StartImagePipelineExecution
	générateur d'images : StartResourceStateUpdate
	générateur d'images : UpdateDistributionConfiguration
	générateur d'images : UpdateImagePipeline

Préfixe de service	Actions
	générateur d'images : UpdateInfrastructureConfiguration

Préfixe de service	Actions
inspector	inspecteur : AddAttributesToFindings inspecteur : CreateAssessmentTarget inspecteur : CreateAssessmentTemplate inspecteur : CreateExclusionsPreview inspecteur : CreateResourceGroup inspecteur : DeleteAssessmentRun inspecteur : DeleteAssessmentTarget inspecteur : DeleteAssessmentTemplate inspecteur : DescribeAssessmentRuns inspecteur : DescribeAssessmentTargets inspecteur : DescribeAssessmentTemplates inspecteur : DescribeCrossAccountAccessRole inspecteur : DescribeExclusions inspecteur : DescribeFindings inspecteur : DescribeResourceGroups inspecteur : DescribeRulesPackages inspecteur : GetAssessmentReport inspecteur : GetExclusionsPreview inspecteur : GetTelemetryMetadata inspecteur : ListAssessmentRunAgents inspecteur : ListAssessmentRuns

Préfixe de service	Actions
	inspecteur : ListAssessmentTargets inspecteur : ListAssessmentTemplates inspecteur : ListEventSubscriptions inspecteur : ListExclusions inspecteur : ListFindings inspecteur : ListRulesPackages inspecteur : PreviewAgents inspecteur : RegisterCrossAccountAccessRole inspecteur : RemoveAttributesFromFindings inspecteur : StartAssessmentRun inspecteur : StopAssessmentRun inspecteur : SubscribeToEvent inspecteur : UnsubscribeFromEvent inspecteur : UpdateAssessmentTarget

Préfixe de service	Actions
inspector2	inspecteur 2 : AssociateMember
	inspecteur 2 : BatchGetAccountStatus
	inspecteur 2 : BatchGetCodeSnippet
	inspecteur 2 : BatchGetFindingDetails
	inspecteur 2 : BatchGetFreeTrialInfo
	inspecteur 2 : BatchGetMemberEc2DeepInspectionStatus
	inspecteur 2 : BatchUpdateMemberEc2DeepInspectionStatus
	inspecteur 2 : CancelFindingsReport
	inspecteur 2 : CancelSbomExport
	inspecteur 2 : CreateCisScanConfiguration
	inspecteur 2 : CreateCodeSecurityIntegration
	inspecteur 2 : CreateConnector
	inspecteur 2 : CreateFilter
	inspecteur 2 : CreateFindingsReport
	inspecteur 2 : CreateSbomExport
	inspecteur 2 : DeleteCisScanConfiguration
	inspecteur 2 : DeleteCodeSecurityIntegration
	inspecteur 2 : DeleteConnector
	inspecteur 2 : DeleteFilter
	inspecteur 2 : DescribeOrganizationConfiguration
	inspector2:Disable

Préfixe de service	Actions
	inspecteur 2 : DisableDelegatedAdminAccount
	inspecteur 2 : DisassociateMember
	inspector2:Enable
	inspecteur 2 : EnableDelegatedAdminAccount
	inspecteur 2 : GetCisScanReport
	inspecteur 2 : GetCisScanResultDetails
	inspecteur 2 : GetClustersForImage
	inspecteur 2 : GetCodeSecurityIntegration
	inspecteur 2 : GetCodeSecurityScan
	inspecteur 2 : GetConfiguration
	inspecteur 2 : GetDelegatedAdminAccount
	inspecteur 2 : GetEc2DeepInspectionConfiguration
	inspecteur 2 : GetEncryptionKey
	inspecteur 2 : GetFindingsReportStatus
	inspecteur 2 : GetMember
	inspecteur 2 : GetSbomExport
	inspecteur 2 : ListAccountPermissions
	inspecteur 2 : ListCisScanConfigurations
	inspecteur 2 : ListCisScanResultsAggregatedByChecks
	inspecteur 2 : ListCisScanResultsAggregatedByTargetResource
	inspecteur 2 : ListCisScans

Préfixe de service	Actions
	inspecteur 2 : ListCodeSecurityIntegrations
	inspecteur 2 : ListCodeSecurityScanConfigurations
	inspecteur 2 : ListConnectorScanConfigurations
	inspecteur 2 : ListConnectors
	inspecteur 2 : ListCoverage
	inspecteur 2 : ListCoverageStatistics
	inspecteur 2 : ListDelegatedAdminAccounts
	inspecteur 2 : ListFilters
	inspecteur 2 : ListFindingAggregations
	inspecteur 2 : ListFindings
	inspecteur 2 : ListMembers
	inspecteur 2 : ListUsageTotals
	inspecteur 2 : ResetEncryptionKey
	inspecteur 2 : SearchVulnerabilities
	inspecteur 2 : SendCisSessionHealth
	inspecteur 2 : SendCisSessionTelemetry
	inspecteur 2 : StartCisSession
	inspecteur 2 : StartCodeSecurityScan
	inspecteur 2 : StopCisSession
	inspecteur 2 : UpdateCisScanConfiguration
	inspecteur 2 : UpdateCodeSecurityIntegration

Préfixe de service	Actions
	inspecteur 2 : UpdateConfiguration
	inspecteur 2 : UpdateConnector
	inspecteur 2 : UpdateEc2DeepInspectionConfiguration
	inspecteur 2 : UpdateEncryptionKey
	inspecteur 2 : UpdateFilter
	inspecteur 2 : UpdateOrgEc2DeepInspectionConfiguration
	inspecteur 2 : UpdateOrganizationConfiguration

Préfixe de service	Actions
iot	IoT : AcceptCertificateTransfer IoT : AddThingToBillingGroup IoT : AddThingToThingGroup IoT : AssociateSbomWithPackageVersion IoT : AssociateTargetsWithJob IoT : AttachPolicy IoT : AttachPrincipalPolicy IoT : AttachSecurityProfile IoT : AttachThingPrincipal IoT : CancelAuditMitigationActionsTask IoT : CancelAuditTask IoT : CancelCertificateTransfer IoT : CancelDetectMitigationActionsTask IoT : CancelJob IoT : CancelJobExecution IoT : ClearDefaultAuthorizer IoT : ConfirmTopicRuleDestination IoT : CreateAuditSuppression IoT : CreateAuthorizer IoT : CreateBillingGroup IoT : CreateCertificateFromCsr

Préfixe de service	Actions
	IoT : CreateCertificateProvider IoT : CreateCommand IoT : CreateCustomMetric IoT : CreateDimension IoT : CreateDomainConfiguration IoT : CreateDynamicThingGroup IoT : CreateFleetMetric IoT : CreateJob IoT : CreateJobTemplate IoT : CreateKeysAndCertificate IoT : CreateMitigationAction iot:CreateOTAUpdate IoT : CreatePackage IoT : CreatePackageVersion IoT : CreatePolicy IoT : CreatePolicyVersion IoT : CreateProvisioningClaim IoT : CreateProvisioningTemplate IoT : CreateProvisioningTemplateVersion IoT : CreateRoleAlias IoT : CreateScheduledAudit

Préfixe de service	Actions
	IoT : CreateSecurityProfile IoT : CreateStream IoT : CreateThing IoT : CreateThingGroup IoT : CreateThingType IoT : CreateTopicRule IoT : CreateTopicRuleDestination IoT : DeleteAccountAuditConfiguration IoT : DeleteAuditSuppression IoT : DeleteAuthorizer IoT : DeleteBillingGroup iot:DeleteCACertificate IoT : DeleteCertificate IoT : DeleteCertificateProvider IoT : DeleteCommand IoT : DeleteCustomMetric IoT : DeleteDimension IoT : DeleteDomainConfiguration IoT : DeleteDynamicThingGroup IoT : DeleteFleetMetric IoT : DeleteJob

Préfixe de service	Actions
	IoT : DeleteJobExecution IoT : DeleteJobTemplate IoT : DeleteMitigationAction iot:DeleteOTAUpdate IoT : DeletePackage IoT : DeletePackageVersion IoT : DeletePolicy IoT : DeletePolicyVersion IoT : DeleteProvisioningTemplate IoT : DeleteProvisioningTemplateVersion IoT : DeleteRegistrationCode IoT : DeleteRoleAlias IoT : DeleteScheduledAudit IoT : DeleteSecurityProfile IoT : DeleteStream IoT : DeleteThing IoT : DeleteThingGroup IoT : DeleteThingType IoT : DeleteTopicRule IoT : DeleteTopicRuleDestination IoT : DeleteV2LoggingLevel

Préfixe de service	Actions
	IoT : DeprecateThingType IoT : DescribeAccountAuditConfiguration IoT : DescribeAuditFinding IoT : DescribeAuditMitigationActionsTask IoT : DescribeAuditSuppression IoT : DescribeAuditTask IoT : DescribeAuthorizer IoT : DescribeBillingGroup iot:DescribeCACertificate IoT : DescribeCertificate IoT : DescribeCertificateProvider IoT : DescribeCustomMetric IoT : DescribeDefaultAuthorizer IoT : DescribeDetectMitigationActionsTask IoT : DescribeDimension IoT : DescribeDomainConfiguration IoT : DescribeEncryptionConfiguration IoT : DescribeEndpoint IoT : DescribeEventConfigurations IoT : DescribeFleetMetric IoT : DescribeIndex

Préfixe de service	Actions
	IoT : DescribeJob IoT : DescribeJobExecution IoT : DescribeJobTemplate IoT : DescribeManagedJobTemplate IoT : DescribeMitigationAction IoT : DescribeProvisioningTemplate IoT : DescribeProvisioningTemplateVersion IoT : DescribeRoleAlias IoT : DescribeScheduledAudit IoT : DescribeSecurityProfile IoT : DescribeStream IoT : DescribeThing IoT : DescribeThingGroup IoT : DescribeThingRegistrationTask IoT : DescribeThingType IoT : DetachPolicy IoT : DetachPrincipalPolicy IoT : DetachSecurityProfile IoT : DetachThingPrincipal IoT : DisableTopicRule IoT : DisassociateSbomFromPackageVersion

Préfixe de service	Actions
	IoT : EnableTopicRule IoT : GetBehaviorModelTrainingSummaries IoT : GetBucketsAggregation IoT : GetCardinality IoT : GetCommand IoT : GetEffectivePolicies IoT : GetJobDocument IoT : GetLoggingOptions iot:GetOTAUpdate IoT : GetPackage IoT : GetPackageConfiguration IoT : GetPackageVersion IoT : GetPercentiles IoT : GetPolicy IoT : GetPolicyVersion IoT : GetRegistrationCode IoT : GetStatistics IoT : GetThingConnectivityData IoT : GetTopicRule IoT : GetTopicRuleDestination IoT : GetV2LoggingOptions

Préfixe de service	Actions
	IoT : ListActiveViolations IoT : ListAttachedPolicies IoT : ListAuditFindings IoT : ListAuditMitigationActionsExecutions IoT : ListAuditMitigationActionsTasks IoT : ListAuditSuppressions IoT : ListAuditTasks IoT : ListAuthorizers IoT : ListBillingGroups iot:ListCACertificates IoT : ListCertificateProviders IoT : ListCertificates IoT : ListCertificatesByCA IoT : ListCommands IoT : ListCustomMetrics IoT : ListDetectMitigationActionsExecutions IoT : ListDetectMitigationActionsTasks IoT : ListDimensions IoT : ListDomainConfigurations IoT : ListFleetMetrics IoT : ListIndices

Préfixe de service	Actions
	IoT : ListJobExecutionsForJob IoT : ListJobExecutionsForThing IoT : ListJobTemplates IoT : ListJobs IoT : ListManagedJobTemplates IoT : ListMetricValues IoT : ListMitigationActions iot:ListOTAUpdates IoT : ListOutgoingCertificates IoT : ListPackageVersions IoT : ListPackages IoT : ListPolicies IoT : ListPolicyPrincipals IoT : ListPolicyVersions IoT : ListPrincipalPolicies IoT : ListPrincipalThings IoT : ListPrincipalThingsV2 IoT : ListProvisioningTemplateVersions IoT : ListProvisioningTemplates IoT : ListRelatedResourcesForAuditFinding IoT : ListRoleAliases

Préfixe de service	Actions
	IoT : ListSbomValidationResults
	IoT : ListScheduledAudits
	IoT : ListSecurityProfiles
	IoT : ListSecurityProfilesForTarget
	IoT : ListStreams
	IoT : ListTargetsForPolicy
	IoT : ListTargetsForSecurityProfile
	IoT : ListThingGroups
	IoT : ListThingGroupsForThing
	IoT : ListThingPrincipals
	IoT : ListThingPrincipalsV2
	IoT : ListThingRegistrationTaskReports
	IoT : ListThingRegistrationTasks
	IoT : ListThingTypes
	IoT : ListThings
	IoT : ListThingsInBillingGroup
	IoT : ListThingsInThingGroup
	IoT : ListTopicRuleDestinations
	IoT : ListTopicRules
	IoT : ListV2LoggingLevels
	IoT : ListViolationEvents

Préfixe de service	Actions
	IoT : PutVerificationStateOnViolation iot:RegisterCACertificate IoT : RegisterCertificate IoT : RegisterCertificateWithoutCA IoT : RegisterThing IoT : RejectCertificateTransfer IoT : RemoveThingFromBillingGroup IoT : RemoveThingFromThingGroup IoT : ReplaceTopicRule IoT : SearchIndex IoT : SetDefaultAuthorizer IoT : SetDefaultPolicyVersion IoT : SetLoggingOptions IoT : SetV2LoggingLevel IoT : SetV2LoggingOptions IoT : StartAuditMitigationActionsTask IoT : StartDetectMitigationActionsTask IoT : StartOnDemandAuditTask IoT : StartThingRegistrationTask IoT : StopThingRegistrationTask IoT : TestAuthorization

Préfixe de service	Actions
	IoT : TestInvokeAuthorizer IoT : TransferCertificate IoT : UpdateAccountAuditConfiguration IoT : UpdateAuditSuppression IoT : UpdateAuthorizer IoT : UpdateBillingGroup iot:UpdateCACertificate IoT : UpdateCertificate IoT : UpdateCertificateProvider IoT : UpdateCommand IoT : UpdateCustomMetric IoT : UpdateDimension IoT : UpdateDomainConfiguration IoT : UpdateDynamicThingGroup IoT : UpdateEncryptionConfiguration IoT : UpdateEventConfigurations IoT : UpdateFleetMetric IoT : UpdateIndexingConfiguration IoT : UpdateJob IoT : UpdateMitigationAction IoT : UpdatePackage

Préfixe de service	Actions
	IoT : UpdatePackageConfiguration IoT : UpdatePackageVersion IoT : UpdateProvisioningTemplate IoT : UpdateRoleAlias IoT : UpdateScheduledAudit IoT : UpdateSecurityProfile IoT : UpdateStream IoT : UpdateThing IoT : UpdateThingGroup IoT : UpdateThingGroupsForThing IoT : UpdateThingType IoT : UpdateTopicRuleDestination IoT : ValidateSecurityProfileBehaviors

Préfixe de service	Actions
iotanalytics	Analyse de l'IoT : CancelPipelineReprocessing
	Analyse de l'IoT : CreateChannel
	Analyse de l'IoT : CreateDataset
	Analyse de l'IoT : CreateDatasetContent
	Analyse de l'IoT : CreateDatastore
	Analyse de l'IoT : CreatePipeline
	Analyse de l'IoT : DeleteChannel
	Analyse de l'IoT : DeleteDataset
	Analyse de l'IoT : DeleteDatasetContent
	Analyse de l'IoT : DeleteDatastore
	Analyse de l'IoT : DeletePipeline
	Analyse de l'IoT : DescribeChannel
	Analyse de l'IoT : DescribeDataset
	Analyse de l'IoT : DescribeDatastore
	Analyse de l'IoT : DescribeLoggingOptions
	Analyse de l'IoT : DescribePipeline
	Analyse de l'IoT : GetDatasetContent
	Analyse de l'IoT : ListChannels
	Analyse de l'IoT : ListDatasetContents
	Analyse de l'IoT : ListDatasets
	Analyse de l'IoT : ListDatastores

Préfixe de service	Actions
	Analyse de l'IoT : ListPipelines
	Analyse de l'IoT : PutLoggingOptions
	Analyse de l'IoT : RunPipelineActivity
	Analyse de l'IoT : SampleChannelData
	Analyse de l'IoT : StartPipelineReprocessing
	Analyse de l'IoT : UpdateChannel
	Analyse de l'IoT : UpdateDataset
	Analyse de l'IoT : UpdateDatastore
	Analyse de l'IoT : UpdatePipeline
iotdeviceadvisor	conseiller pour appareils IoT : CreateSuiteDefinition
	conseiller pour appareils IoT : DeleteSuiteDefinition
	conseiller pour appareils IoT : GetEndpoint
	conseiller pour appareils IoT : GetSuiteDefinition
	conseiller pour appareils IoT : GetSuiteRun
	conseiller pour appareils IoT : GetSuiteRunReport
	conseiller pour appareils IoT : ListSuiteDefinitions
	conseiller pour appareils IoT : ListSuiteRuns
	conseiller pour appareils IoT : StartSuiteRun
	conseiller pour appareils IoT : StopSuiteRun
	conseiller pour appareils IoT : UpdateSuiteDefinition

Préfixe de service	Actions
iotevents	événements liés à l'Internet des objets : BatchAcknowledgeAlarm
	événements liés à l'Internet des objets : BatchDeleteDetector
	événements liés à l'Internet des objets : BatchDisableAlarm
	événements liés à l'Internet des objets : BatchEnableAlarm
	événements liés à l'Internet des objets : BatchResetAlarm
	événements liés à l'Internet des objets : BatchSnoozeAlarm
	événements liés à l'Internet des objets : BatchUpdateDetector
	événements liés à l'Internet des objets : CreateAlarmModel
	événements liés à l'Internet des objets : CreateDetectorModel
	événements liés à l'Internet des objets : CreateInput
	événements liés à l'Internet des objets : DeleteAlarmModel
	événements liés à l'Internet des objets : DeleteDetectorModel
	événements liés à l'Internet des objets : DeleteInput
	événements liés à l'Internet des objets : DescribeAlarm
	événements liés à l'Internet des objets : DescribeAlarmModel
	événements liés à l'Internet des objets : DescribeDetector
	événements liés à l'Internet des objets : DescribeDetectorModel
	événements liés à l'Internet des objets : DescribeDetectorMo delAnalysis
	événements liés à l'Internet des objets : DescribeInput
	événements liés à l'Internet des objets : DescribeLoggingOptions

Préfixe de service	Actions
	événements liés à l'Internet des objets : GetDetectorModelAnalysisResults événements liés à l'Internet des objets : ListAlarmModelVersions événements liés à l'Internet des objets : ListAlarmModels événements liés à l'Internet des objets : ListAlarms événements liés à l'Internet des objets : ListDetectorModelVersions événements liés à l'Internet des objets : ListDetectorModels événements liés à l'Internet des objets : ListDetectors événements liés à l'Internet des objets : ListInputRoutings événements liés à l'Internet des objets : ListInputs événements liés à l'Internet des objets : PutLoggingOptions événements liés à l'Internet des objets : StartDetectorModelAnalysis événements liés à l'Internet des objets : UpdateAlarmModel événements liés à l'Internet des objets : UpdateDetectorModel événements liés à l'Internet des objets : UpdateInput
iotfleethub	IoT Fleethub : CreateApplication IoT Fleethub : DeleteApplication IoT Fleethub : DescribeApplication IoT Fleethub : ListApplications IoT Fleethub : UpdateApplication

Préfixe de service	Actions
iotsitewise	à l'extérieur du site : AssociateAssets
	à l'extérieur du site : AssociateTimeSeriesToAssetProperty
	à l'extérieur du site : BatchAssociateProjectAssets
	à l'extérieur du site : BatchDisassociateProjectAssets
	à l'extérieur du site : CreateAccessPolicy
	à l'extérieur du site : CreateAsset
	à l'extérieur du site : CreateAssetModel
	à l'extérieur du site : CreateAssetModelCompositeModel
	à l'extérieur du site : CreateBulkImportJob
	à l'extérieur du site : CreateComputationModel
	à l'extérieur du site : CreateDashboard
	à l'extérieur du site : CreateDataset
	à l'extérieur du site : CreateGateway
	à l'extérieur du site : CreatePortal
	à l'extérieur du site : CreateProject
	à l'extérieur du site : DeleteAccessPolicy
	à l'extérieur du site : DeleteAsset
	à l'extérieur du site : DeleteAssetModel
	à l'extérieur du site : DeleteAssetModelCompositeModel
	à l'extérieur du site : DeleteComputationModel
	à l'extérieur du site : DeleteDashboard

Préfixe de service	Actions
	à l'extérieur du site : DeleteDataset
	à l'extérieur du site : DeleteGateway
	à l'extérieur du site : DeletePortal
	à l'extérieur du site : DeleteProject
	à l'extérieur du site : DeleteTimeSeries
	à l'extérieur du site : DescribeAccessPolicy
	à l'extérieur du site : DescribeAsset
	à l'extérieur du site : DescribeAssetCompositeModel
	à l'extérieur du site : DescribeAssetModel
	à l'extérieur du site : DescribeAssetModelCompositeModel
	à l'extérieur du site : DescribeAssetModelInterfaceRelationship
	à l'extérieur du site : DescribeAssetProperty
	à l'extérieur du site : DescribeBulkImportJob
	à l'extérieur du site : DescribeComputationModel
	à l'extérieur du site : DescribeComputationModelExecutionSummary
	à l'extérieur du site : DescribeDashboard
	à l'extérieur du site : DescribeDataset
	à l'extérieur du site : DescribeDefaultEncryptionConfiguration
	à l'extérieur du site : DescribeExecution
	à l'extérieur du site : DescribeGateway
	à l'extérieur du site : DescribeGatewayCapabilityConfiguration

Préfixe de service	Actions
	à l'extérieur du site : DescribeLoggingOptions
	à l'extérieur du site : DescribePortal
	à l'extérieur du site : DescribeProject
	à l'extérieur du site : DescribeStorageConfiguration
	à l'extérieur du site : DescribeTimeSeries
	à l'extérieur du site : DisassociateAssets
	à l'extérieur du site : DisassociateTimeSeriesFromAssetProperty
	à l'extérieur du site : ExecuteAction
	à l'extérieur du site : ExecuteQuery
	à l'extérieur du site : ListAccessPolicies
	à l'extérieur du site : ListActions
	à l'extérieur du site : ListAssetModelCompositeModels
	à l'extérieur du site : ListAssetModelProperties
	à l'extérieur du site : ListAssetModels
	en dehors du site : ListAssetProperties
	en dehors du site : ListAssetRelationships
	en dehors du site : ListAssets
	en dehors du site : ListAssociatedAssets
	en dehors du site : ListBulkImportJobs
	en dehors du site : ListCompositionRelationships
	en dehors du site : ListComputationModelDataBindingUsages

Préfixe de service	Actions
	en dehors du site : ListComputationModelResolveToResources
	en dehors du site : ListComputationModels
	en dehors du site : ListDashboards
	en dehors du site : ListDatasets
	en dehors du site : ListExecutions
	en dehors du site : ListGateways
	en dehors du site : ListInterfaceRelationships
	en dehors du site : ListPortals
	en dehors du site : ListProjectAssets
	en dehors du site : ListProjects
	en dehors du site : ListTimeSeries
	en dehors du site : PutDefaultEncryptionConfiguration
	en dehors du site : PutLoggingOptions
	en dehors du site : PutStorageConfiguration
	en dehors du site : UpdateAccessPolicy
	en dehors du site : UpdateAsset
	en dehors du site : UpdateAssetModel
	en dehors du site : UpdateAssetModelCompositeModel
	en dehors du site : UpdateAssetProperty
	en dehors du site : UpdateComputationModel
	en dehors du site : UpdateDashboard

Préfixe de service	Actions
	en dehors du site : UpdateDataset en dehors du site : UpdateGateway en dehors du site : UpdateGatewayCapabilityConfiguration en dehors du site : UpdatePortal en dehors du site : UpdateProject

Préfixe de service	Actions
iottwinmaker	iottwinmaker : CancelMetadataTransferJob iottwinmaker : CreateComponentType iottwinmaker : CreateEntity iottwinmaker : CreateMetadataTransferJob iottwinmaker : CreateScene iottwinmaker : CreateSyncJob iottwinmaker : CreateWorkspace iottwinmaker : DeleteComponentType iottwinmaker : DeleteEntity iottwinmaker : DeleteScene iottwinmaker : DeleteSyncJob iottwinmaker : DeleteWorkspace iottwinmaker : ExecuteQuery iottwinmaker : GetMetadataTransferJob iottwinmaker : GetPricingPlan iottwinmaker : GetScene iottwinmaker : GetSyncJob iottwinmaker : ListComponentTypes iottwinmaker : ListComponents iottwinmaker : ListEntities iottwinmaker : ListMetadataTransferJobs

Préfixe de service	Actions
	iottwinmaker : ListProperties iottwinmaker : ListScenes iottwinmaker : ListSyncJobs iottwinmaker : ListSyncResources iottwinmaker : ListWorkspaces iottwinmaker : UpdateComponentType iottwinmaker : UpdateEntity iottwinmaker : UpdatePricingPlan iottwinmaker : UpdateScene iottwinmaker : UpdateWorkspace

Préfixe de service	Actions
iotwireless	IoT sans fil : AssociateAwsAccountWithPartnerAccount
	IoT sans fil : AssociateMulticastGroupWithFuotaTask
	IoT sans fil : AssociateWirelessDeviceWithFuotaTask
	IoT sans fil : AssociateWirelessDeviceWithMulticastGroup
	IoT sans fil : AssociateWirelessDeviceWithThing
	IoT sans fil : AssociateWirelessGatewayWithCertificate
	IoT sans fil : AssociateWirelessGatewayWithThing
	IoT sans fil : CancelMulticastGroupSession
	IoT sans fil : CreateDestination
	IoT sans fil : CreateDeviceProfile
	IoT sans fil : CreateFuotaTask
	IoT sans fil : CreateMulticastGroup
	IoT sans fil : CreateNetworkAnalyzerConfiguration
	IoT sans fil : CreateServiceProfile
	IoT sans fil : CreateWirelessDevice
	IoT sans fil : CreateWirelessGateway
	IoT sans fil : CreateWirelessGatewayTask
	IoT sans fil : CreateWirelessGatewayTaskDefinition
	IoT sans fil : DeleteDestination
	IoT sans fil : DeleteDeviceProfile
	IoT sans fil : DeleteFuotaTask

Préfixe de service	Actions
	IoT sans fil : DeleteMulticastGroup
	IoT sans fil : DeleteNetworkAnalyzerConfiguration
	IoT sans fil : DeleteQueuedMessages
	IoT sans fil : DeleteServiceProfile
	IoT sans fil : DeleteWirelessDevice
	IoT sans fil : DeleteWirelessDeviceImportTask
	IoT sans fil : DeleteWirelessGateway
	IoT sans fil : DeleteWirelessGatewayTask
	IoT sans fil : DeleteWirelessGatewayTaskDefinition
	IoT sans fil : DeregisterWirelessDevice
	IoT sans fil : DisassociateAwsAccountFromPartnerAccount
	IoT sans fil : DisassociateMulticastGroupFromFuotaTask
	IoT sans fil : DisassociateWirelessDeviceFromFuotaTask
	IoT sans fil : DisassociateWirelessDeviceFromMulticastGroup
	IoT sans fil : DisassociateWirelessDeviceFromThing
	IoT sans fil : DisassociateWirelessGatewayFromCertificate
	IoT sans fil : DisassociateWirelessGatewayFromThing
	IoT sans fil : GetDestination
	IoT sans fil : GetDeviceProfile
	IoT sans fil : GetEventConfigurationByResourceTypes
	IoT sans fil : GetFuotaTask

Préfixe de service	Actions
	IoT sans fil : GetLogLevelsByResourceTypes
	IoT sans fil : GetMetricConfiguration
	IoT sans fil : GetMetrics
	IoT sans fil : GetMulticastGroup
	IoT sans fil : GetMulticastGroupSession
	IoT sans fil : GetNetworkAnalyzerConfiguration
	IoT sans fil : GetPartnerAccount
	IoT sans fil : GetPosition
	IoT sans fil : GetPositionConfiguration
	IoT sans fil : GetPositionEstimate
	IoT sans fil : GetResourceEventConfiguration
	IoT sans fil : GetResourceLogLevel
	IoT sans fil : GetResourcePosition
	IoT sans fil : GetServiceEndpoint
	IoT sans fil : GetServiceProfile
	IoT sans fil : GetWirelessDevice
	IoT sans fil : GetWirelessDeviceImportTask
	IoT sans fil : GetWirelessDeviceStatistics
	IoT sans fil : GetWirelessGateway
	IoT sans fil : GetWirelessGatewayCertificate
	IoT sans fil : GetWirelessGatewayFirmwareInformation

Préfixe de service	Actions
	IoT sans fil : GetWirelessGatewayStatistics
	IoT sans fil : GetWirelessGatewayTask
	IoT sans fil : GetWirelessGatewayTaskDefinition
	IoT sans fil : ListDestinations
	IoT sans fil : ListDeviceProfiles
	IoT sans fil : ListDevicesForWirelessDeviceImportTask
	IoT sans fil : ListEventConfigurations
	IoT sans fil : ListFuotaTasks
	IoT sans fil : ListMulticastGroups
	IoT sans fil : ListMulticastGroupsByFuotaTask
	IoT sans fil : ListNetworkAnalyzerConfigurations
	IoT sans fil : ListPartnerAccounts
	IoT sans fil : ListPositionConfigurations
	IoT sans fil : ListQueuedMessages
	IoT sans fil : ListServiceProfiles
	IoT sans fil : ListWirelessDeviceImportTasks
	IoT sans fil : ListWirelessDevices
	IoT sans fil : ListWirelessGatewayTaskDefinitions
	IoT sans fil : ListWirelessGateways
	IoT sans fil : PutPositionConfiguration
	IoT sans fil : PutResourceLogLevel

Préfixe de service	Actions
	IoT sans fil : ResetAllResourceLogLevels
	IoT sans fil : ResetResourceLogLevel
	IoT sans fil : SendDataToMulticastGroup
	IoT sans fil : SendDataToWirelessDevice
	IoT sans fil : StartBulkAssociateWirelessDeviceWithMulticastGroup
	IoT sans fil : StartBulkDisassociateWirelessDeviceFromMulticastGroup
	IoT sans fil : StartFuotaTask
	IoT sans fil : StartMulticastGroupSession
	IoT sans fil : StartNetworkAnalyzerStream
	IoT sans fil : StartSingleWirelessDeviceImportTask
	IoT sans fil : StartWirelessDeviceImportTask
	IoT sans fil : TestWirelessDevice
	IoT sans fil : UpdateDestination
	IoT sans fil : UpdateEventConfigurationByResourceTypes
	IoT sans fil : UpdateFuotaTask
	IoT sans fil : UpdateLogLevelsByResourceTypes
	IoT sans fil : UpdateMetricConfiguration
	IoT sans fil : UpdateMulticastGroup
	IoT sans fil : UpdateNetworkAnalyzerConfiguration
	IoT sans fil : UpdatePartnerAccount

Préfixe de service	Actions
	IoT sans fil : UpdatePosition
	IoT sans fil : UpdateResourceEventConfiguration
	IoT sans fil : UpdateResourcePosition
	IoT sans fil : UpdateWirelessDevice
	IoT sans fil : UpdateWirelessDeviceImportTask
	IoT sans fil : UpdateWirelessGateway

Préfixe de service	Actions
ivs	vis : BatchGetChannel vis : BatchGetStreamKey vis : BatchStartViewerSessionRevocation vis : CreateAdConfiguration vis : CreateChannel vis : CreateEncoderConfiguration vis : CreateIngestConfiguration vis : CreateParticipantToken vis : CreatePlaybackRestrictionPolicy vis : CreateRecordingConfiguration vis : CreateStorageConfiguration vis : CreateStreamKey vis : DeleteAdConfiguration vis : DeleteChannel vis : DeleteEncoderConfiguration vis : DeleteIngestConfiguration vis : DeletePlaybackKeyPair vis : DeletePlaybackRestrictionPolicy vis : DeletePublicKey vis : DeleteRecordingConfiguration vis : DeleteStorageConfiguration

Préfixe de service	Actions
	vis : DeleteStreamKey vis : DisconnectParticipant vis : GetAdConfiguration vis : GetChannel vis : GetComposition vis : GetEncoderConfiguration vis : GetIngestConfiguration vis : GetParticipant vis : GetPlaybackKeyPair vis : GetPlaybackRestrictionPolicy vis : GetPublicKey vis : GetRecordingConfiguration vis : GetStorageConfiguration vis : GetStream vis : GetStreamKey vis : GetStreamSession vis : ImportPlaybackKeyPair vis : ImportPublicKey vis : InsertAdBreak vis : ListAdConfigurations vis : ListChannels

Préfixe de service	Actions
	vis : ListCompositions vis : ListEncoderConfigurations vis : ListIngestConfigurations vis : ListParticipantEvents vis : ListParticipantReplicas vis : ListParticipants vis : ListPlaybackKeyPairs vis : ListPlaybackRestrictionPolicies vis : ListPublicKeys vis : ListRecordingConfigurations vis : ListStorageConfigurations vis : ListStreamKeys vis : ListStreamSessions vis : ListStreams vis : PutMetadata vis : StartComposition vis : StartViewerSessionRevocation vis : StopComposition vis : StopStream vis : UpdateAdConfiguration vis : UpdateChannel

Préfixe de service	Actions
	vis : UpdateIngestConfiguration vis : UpdatePlaybackRestrictionPolicy
ivschat	visa : CreateChatToken visa : CreateLoggingConfiguration visa : CreateRoom visa : DeleteLoggingConfiguration visa : DeleteMessage visa : DeleteRoom visa : DisconnectUser visa : GetLoggingConfiguration visa : GetRoom visa : ListLoggingConfigurations visa : ListRooms visa : SendEvent visa : UpdateLoggingConfiguration visa : UpdateRoom

Préfixe de service	Actions
kafka	Kafka : BatchAssociateScramSecret Kafka : BatchDisassociateScramSecret Kafka : CreateCluster Source : V2 CreateCluster Kafka : CreateConfiguration Kafka : CreateReplicator Kafka : CreateVpcConnection Kafka : DeleteCluster Kafka : DeleteClusterPolicy Kafka : DeleteConfiguration Kafka : DeleteReplicator Kafka : DeleteVpcConnection Kafka : DescribeCluster Kafka : DescribeClusterOperation Kafka : DescribeClusterOperationV2 Source : V2 DescribeCluster Kafka : DescribeConfiguration Kafka : DescribeConfigurationRevision Kafka : DescribeVpcConnection Kafka : GetBootstrapBrokers Kafka : GetClusterPolicy

Préfixe de service	Actions
	Kafka : GetCompatibleKafkaVersions Kafka : ListClientVpcConnections Kafka : ListClusterOperations Kafka : ListClusterOperationsV2 Kafka : ListClusters Source : V2 ListClusters Kafka : ListConfigurationRevisions Kafka : ListConfigurations Kafka : ListKafkaVersions Kafka : ListNodes Kafka : ListReplicators Kafka : ListScramSecrets Kafka : ListVpcConnections Kafka : PutClusterPolicy Kafka : RebootBroker Kafka : RejectClientVpcConnection Kafka : UpdateBrokerCount Kafka : UpdateBrokerStorage Kafka : UpdateBrokerType Kafka : UpdateClusterConfiguration Kafka : UpdateClusterKafkaVersion

Préfixe de service	Actions
	Kafka : UpdateConfiguration Kafka : UpdateConnectivity Kafka : UpdateMonitoring Kafka : UpdateRebalancing Kafka : UpdateReplicationInfo Kafka : UpdateSecurity Kafka : UpdateStorage
kafkaconnect	Kafka Connect : CreateConnector Kafka Connect : CreateCustomPlugin Kafka Connect : CreateWorkerConfiguration Kafka Connect : DeleteConnector Kafka Connect : DeleteCustomPlugin Kafka Connect : DeleteWorkerConfiguration Kafka Connect : DescribeConnector Kafka Connect : DescribeCustomPlugin Kafka Connect : DescribeWorkerConfiguration Kafka Connect : ListConnectorOperations Kafka Connect : ListConnectors Kafka Connect : ListCustomPlugins Kafka Connect : ListWorkerConfigurations Kafka Connect : UpdateConnector

Préfixe de service	Actions
kendra	kendra : AssociateEntitiesToExperience kendra : AssociatePersonasToEntities kendra : BatchDeleteDocument kendra : BatchDeleteFeaturedResultsSet kendra : BatchGetDocumentStatus kendra : BatchPutDocument kendra : ClearQuerySuggestions kendra : CreateAccessControlConfiguration kendra : CreateDataSource kendra : CreateExperience kendra : CreateFaq kendra : CreateFeaturedResultsSet kendra : CreateIndex kendra : CreateQuerySuggestionsBlockList kendra : CreateThesaurus kendra : DeleteDataSource kendra : DeleteExperience kendra : DeleteFaq kendra : DeleteIndex kendra : DeletePrincipalMapping kendra : DeleteQuerySuggestionsBlockList

Préfixe de service	Actions
	kendra : DeleteThesaurus kendra : DescribeAccessControlConfiguration kendra : DescribeDataSource kendra : DescribeExperience kendra : DescribeFaq kendra : DescribeFeaturedResultsSet kendra : DescribeIndex kendra : DescribePrincipalMapping kendra : DescribeQuerySuggestionsBlockList kendra : DescribeQuerySuggestionsConfig kendra : DescribeThesaurus kendra : DisassociateEntitiesFromExperience kendra : DisassociatePersonasFromEntities kendra : GetQuerySuggestions kendra : GetSnapshots kendra : ListAccessControlConfigurations kendra : ListDataSourceSyncJobs kendra : ListDataSources kendra : ListEntityPersonas kendra : ListExperienceEntities kendra : ListExperiences

Préfixe de service	Actions
	kendra : ListFaqs kendra : ListFeaturedResultsSets kendra : ListGroupsOlderThanOrderingId kendra : ListIndices kendra : ListQuerySuggestionsBlockLists kendra : ListThesauri kendra : PutPrincipalMapping kendra:Query kendra:Retrieve kendra : StartDataSourceSyncJob kendra : StopDataSourceSyncJob kendra : SubmitFeedback kendra : UpdateDataSource kendra : UpdateExperience kendra : UpdateFeaturedResultsSet kendra : UpdateIndex kendra : UpdateQuerySuggestionsBlockList kendra : UpdateQuerySuggestionsConfig kendra : UpdateThesaurus

Préfixe de service	Actions
kinesis	kinésie : CreateStream
	kinésie : DecreaseStreamRetentionPeriod
	kinésie : DeleteStream
	kinésie : DeregisterStreamConsumer
	kinésie : DescribeAccountSettings
	kinésie : DescribeLimits
	kinésie : DescribeStream
	kinésie : DescribeStreamConsumer
	kinésie : DescribeStreamSummary
	kinésie : DisableEnhancedMonitoring
	kinésie : EnableEnhancedMonitoring
	kinésie : IncreaseStreamRetentionPeriod
	kinésie : ListShards
	kinésie : ListStreamConsumers
	kinésie : ListStreams
	kinésie : MergeShards
	kinésie : RegisterStreamConsumer
	kinésie : SplitShard
	kinésie : StartStreamEncryption
	kinésie : StopStreamEncryption
	kinésie : UpdateAccountSettings

Préfixe de service	Actions
	kinésie : UpdateShardCount kinésie : UpdateStreamMode

Préfixe de service	Actions
kinesisanalytics	Kinesis Analytics : AddApplicationCloudWatchLoggingOption Kinesis Analytics : AddApplicationInput Kinesis Analytics : AddApplicationInputProcessingConfiguration Kinesis Analytics : AddApplicationOutput Kinesis Analytics : AddApplicationReferenceDataSource Kinesis Analytics : AddApplicationVpcConfiguration Kinesis Analytics : CreateApplication Kinesis Analytics : CreateApplicationPresignedUrl Kinesis Analytics : CreateApplicationSnapshot Kinesis Analytics : DeleteApplication Kinesis Analytics : DeleteApplicationCloudWatchLoggingOption Kinesis Analytics : DeleteApplicationInputProcessingConfiguration Kinesis Analytics : DeleteApplicationOutput Kinesis Analytics : DeleteApplicationReferenceDataSource Kinesis Analytics : DeleteApplicationSnapshot Kinesis Analytics : DeleteApplicationVpcConfiguration Kinesis Analytics : DescribeApplication Kinesis Analytics : DescribeApplicationOperation Kinesis Analytics : DescribeApplicationSnapshot Kinesis Analytics : DescribeApplicationVersion Kinesis Analytics : DiscoverInputSchema

Préfixe de service	Actions
	Kinesis Analytics : ListApplicationOperations Kinesis Analytics : ListApplicationSnapshots Kinesis Analytics : ListApplicationVersions Kinesis Analytics : ListApplications Kinesis Analytics : RollbackApplication Kinesis Analytics : StartApplication Kinesis Analytics : StopApplication Kinesis Analytics : UpdateApplication Kinesis Analytics : UpdateApplicationMaintenanceConfiguration

Préfixe de service	Actions
kms	km : CancelKeyDeletion
	km : ConnectCustomKeyStore
	km : CreateAlias
	km : CreateCustomKeyStore
	km : CreateGrant
	km : CreateKey
	kms:Decrypt
	km : DeleteAlias
	km : DeleteCustomKeyStore
	km : DeleteImportedKeyMaterial
	km : DeriveSharedSecret
	km : DescribeCustomKeyStores
	km : DescribeKey
	km : DisableKey
	km : DisableKeyRotation
	km : DisconnectCustomKeyStore
	km : EnableKey
	km : EnableKeyRotation
	kms:Encrypt
	km : GenerateDataKey
	km : GenerateDataKeyPair

Préfixe de service	Actions
	km : GenerateDataKeyPairWithoutPlaintext
	km : GenerateDataKeyWithoutPlaintext
	km : GenerateMac
	km : GenerateRandom
	km : GetKeyLastUsage
	km : GetKeyPolicy
	km : GetKeyRotationStatus
	km : GetParametersForImport
	km : GetPublicKey
	km : ImportKeyMaterial
	km : ListAliases
	km : ListGrants
	km : ListKeyPolicies
	km : ListKeyRotations
	km : ListKeys
	km : ListRetirableGrants
	km : ReplicateKey
	km : RetireGrant
	km : RevokeGrant
	km : RotateKeyOnDemand
	km : ScheduleKeyDeletion

Préfixe de service	Actions
	kms:Sign km : UpdateAlias km : UpdateCustomKeyStore km : UpdateKeyDescription km : UpdatePrimaryRegion kms:Verify km : VerifyMac

Préfixe de service	Actions
lambda	lambda : AddLayerVersionPermission lambda : AddPermission lambda : CreateAlias lambda : CreateCodeSigningConfig lambda : CreateEventSourceMapping lambda : CreateFunction lambda : CreateFunctionUrlConfig lambda : DeleteAlias lambda : DeleteCapacityProvider lambda : DeleteCodeSigningConfig lambda : DeleteEventSourceMapping lambda : DeleteFunction lambda : DeleteFunctionCodeSigningConfig lambda : DeleteFunctionConcurrency lambda : DeleteFunctionEventInvokeConfig lambda : DeleteFunctionUrlConfig lambda : DeleteLayerVersion lambda : DeleteProvisionedConcurrencyConfig lambda : GetAccountSettings lambda : GetAlias lambda : GetCapacityProvider

Préfixe de service	Actions
	lambda : GetCodeSigningConfig lambda : GetEventSourceMapping lambda : GetFunction lambda : GetFunctionCodeSigningConfig lambda : GetFunctionConcurrency lambda : GetFunctionConfiguration lambda : GetFunctionEventInvokeConfig lambda : GetFunctionRecursionConfig lambda : GetFunctionScalingConfig lambda : GetFunctionUrlConfig lambda : GetLayerVersion lambda : GetLayerVersionPolicy lambda : GetPolicy lambda : GetProvisionedConcurrencyConfig lambda : GetRuntimeManagementConfig lambda : ListAliases lambda : ListCapacityProviders lambda : ListCodeSigningConfigs lambda : ListDurableExecutionsByFunction lambda : ListEventSourceMappings lambda : ListFunctionEventInvokeConfigs

Préfixe de service	Actions
	lambda : ListFunctionUrlConfigs lambda : ListFunctions lambda : ListFunctionsByCodeSigningConfig lambda : ListLayerVersions lambda : ListLayers lambda : ListProvisionedConcurrencyConfigs lambda : ListVersionsByFunction lambda : PublishLayerVersion lambda : PublishVersion lambda : PutFunctionCodeSigningConfig lambda : PutFunctionConcurrency lambda : PutFunctionEventInvokeConfig lambda : PutFunctionRecursionConfig lambda : PutFunctionScalingConfig lambda : PutProvisionedConcurrencyConfig lambda : PutRuntimeManagementConfig lambda : RemoveLayerVersionPermission lambda : RemovePermission lambda : UpdateAlias lambda : UpdateCapacityProvider lambda : UpdateCodeSigningConfig

Préfixe de service	Actions
	lambda : UpdateEventSourceMapping lambda : UpdateFunctionCode lambda : UpdateFunctionConfiguration lambda : UpdateFunctionEventInvokeConfig lambda : UpdateFunctionUrlConfig

Préfixe de service	Actions
lex	Alex : BatchCreateCustomVocabularyItem
	Alex : BatchDeleteCustomVocabularyItem
	Alex : BatchUpdateCustomVocabularyItem
	Alex : BuildBotLocale
	Alex : CreateBotAlias
	Alex : CreateBotReplica
	Alex : CreateBotVersion
	Alex : CreateExport
	Alex : CreateIntentVersion
	Alex : CreateResourcePolicy
	Alex : CreateSlotTypeVersion
	Alex : CreateTestSetDiscrepancyReport
	Alex : CreateUploadUrl
	Alex : DeleteBot
	Alex : DeleteBotAnalyzerRecommendation
	Alex : DeleteBotChannelAssociation
	Alex : DeleteBotReplica
	Alex : DeleteExport
	Alex : DeleteImport
	Alex : DeleteIntentVersion
	Alex : DeleteResourcePolicy

Préfixe de service	Actions
	Alex : DeleteSlotTypeVersion
	Alex : DeleteTestSet
	Alex : DeleteUtterances
	Alex : DescribeBotAlias
	Alex : DescribeBotAnalyzerRecommendation
	Alex : DescribeBotRecommendation
	Alex : DescribeBotReplica
	Alex : DescribeBotResourceGeneration
	Alex : DescribeBotVersion
	Alex : DescribeCustomVocabularyMetadata
	Alex : DescribeExport
	Alex : DescribeImport
	Alex : DescribeResourcePolicy
	Alex : DescribeTestExecution
	Alex : DescribeTestSet
	Alex : DescribeTestSetDiscrepancyReport
	Alex : DescribeTestSetGeneration
	Alex : GenerateBotElement
	Alex : GetBot
	Alex : GetBotAlias
	Alex : GetBotAliases

Préfixe de service	Actions
	Alex : GetBotChannelAssociation
	Alex : GetBotChannelAssociations
	Alex : GetBotVersions
	Alex : GetBots
	Alex : GetBuiltinIntent
	Alex : GetBuiltinIntents
	Alex : GetBuiltinSlotTypes
	Alex : GetExport
	Alex : GetImport
	Alex : GetIntent
	Alex : GetIntentVersions
	Alex : GetIntents
	Alex : GetMigration
	Alex : GetMigrations
	Alex : GetSlotType
	Alex : GetSlotTypeVersions
	Alex : GetSlotTypes
	Alex : GetTestExecutionArtifactsUrl
	Alex : GetUtterancesView
	Alex : ListBotAliasReplicas
	Alex : ListBotAliases

Préfixe de service	Actions
	Alex : ListBotRecommendations
	Alex : ListBotReplicas
	Alex : ListBotResourceGenerations
	Alex : ListBotVersionReplicas
	Alex : ListBotVersions
	Alex : ListBots
	Alex : ListBuiltInIntents
	Alex : ListBuiltInSlotTypes
	Alex : ListCustomVocabularyItems
	Alex : ListExports
	Alex : ListImports
	Alex : ListIntentMetrics
	Alex : ListIntentPaths
	Alex : ListRecommendedIntents
	Alex : ListSessionAnalyticsData
	Alex : ListSessionMetrics
	Alex : ListTestExecutionResultItems
	Alex : ListTestExecutions
	Alex : ListTestSets
	Alex : PutBot
	Alex : PutBotAlias

Préfixe de service	Actions
	Alex : PutIntent
	Alex : PutSlotType
	Alex : SearchAssociatedTranscripts
	Alex : StartBotAnalyzer
	Alex : StartBotRecommendation
	Alex : StartImport
	Alex : StartMigration
	Alex : StartTestExecution
	Alex : StartTestSetGeneration
	Alex : StopBotAnalyzer
	Alex : StopBotRecommendation
	Alex : UpdateBotAlias
	Alex : UpdateBotRecommendation
	Alex : UpdateExport
	Alex : UpdateResourcePolicy

Préfixe de service	Actions
license-manager-linux-subscriptions	License-Manager-Linux-Subscriptions : DeregisterSubscriptionProvider License-Manager-Linux-Subscriptions : GetRegisteredSubscriptionProvider License-Manager-Linux-Subscriptions : GetServiceSettings License-Manager-Linux-Subscriptions : ListLinuxSubscriptionInstances License-Manager-Linux-Subscriptions : ListLinuxSubscriptions License-Manager-Linux-Subscriptions : ListRegisteredSubscriptionProviders License-Manager-Linux-Subscriptions : RegisterSubscriptionProvider License-Manager-Linux-Subscriptions : UpdateServiceSettings

Préfixe de service	Actions
lightsail	voile lumineuse : AllocateStaticIp
	voile lumineuse : AttachCertificateToDistribution
	voile lumineuse : AttachDisk
	voile lumineuse : AttachInstancesToLoadBalancer
	voile lumineuse : AttachLoadBalancerTlsCertificate
	voile lumineuse : AttachStaticIp
	voile lumineuse : CloseInstancePublicPorts
	voile lumineuse : CopySnapshot
	voile lumineuse : CreateBucket
	voile lumineuse : CreateBucketAccessKey
	voile lumineuse : CreateCertificate
	voile lumineuse : CreateCloudFormationStack
	voile lumineuse : CreateContactMethod
	voile lumineuse : CreateContainerService
	voile lumineuse : CreateContainerServiceDeployment
	voile lumineuse : CreateContainerServiceRegistryLogin
	voile lumineuse : CreateDisk
	voile lumineuse : CreateDiskFromSnapshot
	voile lumineuse : CreateDiskSnapshot
	voile lumineuse : CreateDistribution
	voile lumineuse : CreateDomain

Préfixe de service	Actions
	voile lumineuse : CreateGUISessionAccessDetails
	voile lumineuse : CreateInstanceSnapshot
	voile lumineuse : CreateInstances
	voile lumineuse : CreateInstancesFromSnapshot
	voile lumineuse : CreateKeyPair
	voile lumineuse : CreateLoadBalancer
	voile lumineuse : CreateLoadBalancerTlsCertificate
	voile lumineuse : CreateRelationalDatabase
	voile lumineuse : CreateRelationalDatabaseFromSnapshot
	voile lumineuse : CreateRelationalDatabaseSnapshot
	voile lumineuse : DeleteAlarm
	voile lumineuse : DeleteAutoSnapshot
	voile lumineuse : DeleteBucket
	voile lumineuse : DeleteBucketAccessKey
	voile lumineuse : DeleteCertificate
	voile lumineuse : DeleteContactMethod
	voile lumineuse : DeleteContainerImage
	voile lumineuse : DeleteContainerService
	voile lumineuse : DeleteDisk
	voile lumineuse : DeleteDiskSnapshot
	voile lumineuse : DeleteDistribution

Préfixe de service	Actions
	voile lumineuse : DeleteDomain
	voile lumineuse : DeleteDomainEntry
	voile lumineuse : DeleteInstance
	voile lumineuse : DeleteInstanceSnapshot
	voile lumineuse : DeleteKeyPair
	voile lumineuse : DeleteKnownHostKeys
	voile lumineuse : DeleteLoadBalancer
	voile lumineuse : DeleteLoadBalancerTlsCertificate
	voile lumineuse : DeleteRelationalDatabase
	voile lumineuse : DeleteRelationalDatabaseSnapshot
	voile lumineuse : DetachCertificateFromDistribution
	voile lumineuse : DetachDisk
	voile lumineuse : DetachInstancesFromLoadBalancer
	voile lumineuse : DetachStaticIp
	voile lumineuse : DisableAddOn
	voile lumineuse : DownloadDefaultKeyPair
	voile lumineuse : EnableAddOn
	voile lumineuse : ExportSnapshot
	voile lumineuse : GetActiveNames
	voile lumineuse : GetAlarms
	voile lumineuse : GetAutoSnapshots

Préfixe de service	Actions
	voile lumineuse : GetBlueprints
	voile lumineuse : GetBucketAccessKeys
	voile lumineuse : GetBucketBundles
	voile lumineuse : GetBucketMetricData
	voile lumineuse : GetBuckets
	voile lumineuse : GetBundles
	voile lumineuse : GetCertificates
	voile lumineuse : GetCloudFormationStackRecords
	voile lumineuse : GetContactMethods
	voile lumineuse : GetContainerAPIMetadata
	voile lumineuse : GetContainerImages
	voile lumineuse : GetContainerLog
	voile lumineuse : GetContainerServiceDeployments
	voile lumineuse : GetContainerServiceMetricData
	voile lumineuse : GetContainerServicePowers
	voile lumineuse : GetContainerServices
	voile lumineuse : GetCostEstimate
	voile lumineuse : GetDisk
	voile lumineuse : GetDiskSnapshot
	voile lumineuse : GetDiskSnapshots
	voile lumineuse : GetDisks

Préfixe de service	Actions
	voile lumineuse : GetDistributionBundles
	voile lumineuse : GetDistributionLatestCacheReset
	voile lumineuse : GetDistributionMetricData
	voile lumineuse : GetDistributions
	voile lumineuse : GetDomain
	voile lumineuse : GetExportSnapshotRecords
	voile lumineuse : GetInstance
	voile lumineuse : GetInstanceMetricData
	voile lumineuse : GetInstancePortStates
	voile lumineuse : GetInstanceSnapshot
	voile lumineuse : GetInstanceSnapshots
	voile lumineuse : GetInstanceState
	voile lumineuse : GetInstances
	voile lumineuse : GetKeyPair
	voile lumineuse : GetKeyPairs
	voile lumineuse : GetLoadBalancer
	voile lumineuse : GetLoadBalancerMetricData
	voile lumineuse : GetLoadBalancerTlsCertificates
	voile lumineuse : GetLoadBalancerTlsPolicies
	voile lumineuse : GetLoadBalancers
	voile lumineuse : GetOperation

Préfixe de service	Actions
	voile lumineuse : GetOperations
	voile lumineuse : GetOperationsForResource
	voile lumineuse : GetRegions
	voile lumineuse : GetRelationalDatabase
	voile lumineuse : GetRelationalDatabaseBlueprints
	voile lumineuse : GetRelationalDatabaseBundles
	voile lumineuse : GetRelationalDatabaseEvents
	voile lumineuse : GetRelationalDatabaseLogEvents
	voile lumineuse : GetRelationalDatabaseLogStreams
	voile lumineuse : GetRelationalDatabaseMasterUserPassword
	voile lumineuse : GetRelationalDatabaseMetricData
	voile lumineuse : GetRelationalDatabaseParameters
	voile lumineuse : GetRelationalDatabaseSnapshot
	voile lumineuse : GetRelationalDatabaseSnapshots
	voile lumineuse : GetRelationalDatabases
	voile lumineuse : GetSetupHistory
	voile lumineuse : GetStaticIp
	voile lumineuse : GetStaticIps
	voile lumineuse : ImportKeyPair
	voile lumineuse : IsVpcPeered
	voile lumineuse : OpenInstancePublicPorts

Préfixe de service	Actions
	voile lumineuse : PeerVpc
	voile lumineuse : PutAlarm
	voile lumineuse : PutInstancePublicPorts
	voile lumineuse : RebootInstance
	voile lumineuse : RebootRelationalDatabase
	voile lumineuse : RegisterContainerImage
	voile lumineuse : ReleaseStaticIp
	voile lumineuse : ResetDistributionCache
	voile lumineuse : SendContactMethodVerification
	voile lumineuse : SetIpAddressType
	voile lumineuse : SetResourceAccessForBucket
	voile lumineuse : SetupInstanceHttps
	lightsail:StartGUISession
	voile lumineuse : StartInstance
	voile lumineuse : StartRelationalDatabase
	lightsail:StopGUISession
	voile lumineuse : StopInstance
	voile lumineuse : StopRelationalDatabase
	voile lumineuse : TestAlarm
	voile lumineuse : UnpeerVpc
	voile lumineuse : UpdateBucket

Préfixe de service	Actions
	voile lumineuse : UpdateBucketBundle
	voile lumineuse : UpdateContainerService
	voile lumineuse : UpdateDistribution
	voile lumineuse : UpdateDistributionBundle
	voile lumineuse : UpdateDomainEntry
	voile lumineuse : UpdateInstanceMetadataOptions
	voile lumineuse : UpdateLoadBalancerAttribute
	voile lumineuse : UpdateRelationalDatabase
	voile lumineuse : UpdateRelationalDatabaseParameters

Préfixe de service	Actions
journaux	journaux : AssociateKmsKey journaux : AssociateSourceToS3TableIntegration journaux : CancelExportTask journaux : CancelImportTask journaux : CreateDelivery journaux : CreateExportTask journaux : CreateLogAnomalyDetector journaux : CreateLogGroup journaux : CreateLogStream journaux : CreateLookupTable journaux : DeleteDataProtectionPolicy journaux : DeleteDelivery journaux : DeleteDeliveryDestination journaux : DeleteDeliveryDestinationPolicy journaux : DeleteDeliverySource journaux : DeleteDestination journaux : DeleteIndexPolicy journaux : DeleteIntegration journaux : DeleteLogAnomalyDetector journaux : DeleteLogGroup journaux : DeleteLogStream

Préfixe de service	Actions
	journaux : DeleteLookupTable journaux : DeleteMetricFilter journaux : DeleteQueryDefinition journaux : DeleteResourcePolicy journaux : DeleteRetentionPolicy journaux : DeleteScheduledQuery journaux : DeleteSubscriptionFilter journaux : DeleteSyslogConfiguration journaux : DeleteTransformer journaux : DescribeAccountPolicies journaux : DescribeConfigurationTemplates journaux : DescribeDeliveries journaux : DescribeDeliveryDestinations journaux : DescribeDeliverySources journaux : DescribeDestinations journaux : DescribeExportTasks journaux : DescribeFieldIndexes journaux : DescribeImportTaskBatches journaux : DescribeImportTasks journaux : DescribeIndexPolicies journaux : DescribeLogGroups

Préfixe de service	Actions
	journaux : DescribeLogStreams journaux : DescribeLookupTables journaux : DescribeMetricFilters journaux : DescribeQueries journaux : DescribeQueryDefinitions journaux : DescribeResourcePolicies journaux : DescribeSubscriptionFilters journaux : DisassociateKmsKey journaux : DisassociateSourceFromS3TableIntegration journaux : GetDataProtectionPolicy journaux : GetDelivery journaux : GetDeliveryDestination journaux : GetDeliveryDestinationPolicy journaux : GetDeliverySource journaux : GetIntegration journaux : GetLogAnomalyDetector journaux : GetLogFields journaux : GetLogGroupFields journaux : GetLogRecord journaux : GetLookupTable journaux : GetQueryResults

Préfixe de service	Actions
	journaux : GetScheduledQuery journaux : GetScheduledQueryHistory journaux : GetStorageTierPolicy journaux : GetTransformer journaux : ListAnomalies journaux : ListIntegrations journaux : ListLogAnomalyDetectors journaux : ListLogGroupsForQuery journaux : ListScheduledQueries journaux : ListSourcesForS3TableIntegration journaux : ListSyslogConfigurations journaux : PutBearerTokenAuthentication journaux : PutDataProtectionPolicy journaux : PutDeliveryDestination journaux : PutDeliveryDestinationPolicy journaux : PutDeliverySource journaux : PutDestination journaux : PutDestinationPolicy journaux : PutIndexPolicy journaux : PutIntegration journaux : PutLogGroupDeletionProtection

Préfixe de service	Actions
	journaux : PutMetricFilter journaux : PutQueryDefinition journaux : PutResourcePolicy journaux : PutRetentionPolicy journaux : PutStorageTierPolicy journaux : PutSubscriptionFilter journaux : PutSyslogConfiguration journaux : PutTransformer journaux : StartLiveTail journaux : StartQuery journaux : StopQuery journaux : TestMetricFilter journaux : TestTransformer journaux : UpdateAnomaly journaux : UpdateDeliveryConfiguration journaux : UpdateLogAnomalyDetector journaux : UpdateLookupTable

Préfixe de service	Actions
lookoutequipment	équipement de surveillance : CreateDataset
	équipement de surveillance : CreateInferenceScheduler
	équipement de surveillance : CreateLabel
	équipement de surveillance : CreateLabelGroup
	équipement de surveillance : CreateModel
	équipement de surveillance : DeleteDataset
	équipement de surveillance : DeleteInferenceScheduler
	équipement de surveillance : DeleteLabel
	équipement de surveillance : DeleteLabelGroup
	équipement de surveillance : DeleteModel
	équipement de surveillance : DeleteResourcePolicy
	équipement de surveillance : DeleteRetrainingScheduler
	équipement de surveillance : DescribeDataIngestionJob
	équipement de surveillance : DescribeDataset
	équipement de surveillance : DescribeInferenceScheduler
	équipement de surveillance : DescribeLabelGroup
	équipement de surveillance : DescribeModel
	équipement de surveillance : DescribeModelVersion
	équipement de surveillance : DescribeResourcePolicy
	équipement de surveillance : DescribeRetrainingScheduler
	lookoutequipment:Describelabel

Préfixe de service	Actions
	équipement de surveillance : ImportDataset
	équipement de surveillance : ImportModelVersion
	équipement de surveillance : ListDataIngestionJobs
	équipement de surveillance : ListDatasets
	équipement de surveillance : ListInferenceEvents
	équipement de surveillance : ListInferenceExecutions
	équipement de surveillance : ListInferenceSchedulers
	équipement de surveillance : ListLabelGroups
	équipement de surveillance : ListLabels
	équipement de surveillance : ListModelVersions
	équipement de surveillance : ListModels
	équipement de surveillance : ListRetrainingSchedulers
	équipement de surveillance : ListSensorStatistics
	équipement de surveillance : PutResourcePolicy
	équipement de surveillance : StartDataIngestionJob
	équipement de surveillance : StartInferenceScheduler
	équipement de surveillance : StartRetrainingScheduler
	équipement de surveillance : StopInferenceScheduler
	équipement de surveillance : StopRetrainingScheduler
	équipement de surveillance : UpdateActiveModelVersion
	équipement de surveillance : UpdateInferenceScheduler

Préfixe de service	Actions
	équipement de surveillance : UpdateLabelGroup équipement de surveillance : UpdateModel équipement de surveillance : UpdateRetrainingScheduler

Préfixe de service	Actions
lookoutmetrics	Métriques de surveillance : ActivateAnomalyDetector
	Métriques de surveillance : BackTestAnomalyDetector
	Métriques de surveillance : CreateAlert
	Métriques de surveillance : CreateAnomalyDetector
	Métriques de surveillance : CreateMetricSet
	Métriques de surveillance : DeactivateAnomalyDetector
	Métriques de surveillance : DeleteAlert
	Métriques de surveillance : DeleteAnomalyDetector
	Métriques de surveillance : DescribeAlert
	Métriques de surveillance : DescribeAnomalyDetectionExecutions
	Métriques de surveillance : DescribeAnomalyDetector
	Métriques de surveillance : DescribeMetricSet
	Métriques de surveillance : DetectMetricSetConfig
	Métriques de surveillance : GetAnomalyGroup
	Métriques de surveillance : GetDataQualityMetrics
	Métriques de surveillance : GetFeedback
	Métriques de surveillance : GetSampleData
	Métriques de surveillance : ListAlerts
	Métriques de surveillance : ListAnomalyDetectors
	Métriques de surveillance : ListAnomalyGroupRelatedMetrics
	Métriques de surveillance : ListAnomalyGroupSummaries

Préfixe de service	Actions
	Métriques de surveillance : ListAnomalyGroupTimeSeries Métriques de surveillance : ListMetricSets Métriques de surveillance : PutFeedback Métriques de surveillance : UpdateAlert Métriques de surveillance : UpdateAnomalyDetector Métriques de surveillance : UpdateMetricSet

Préfixe de service	Actions
m2	m2 : CancelBatchJobExecution
	m2 : CreateApplication
	m2 : CreateDataSetExportTask
	m2 : CreateDataSetImportTask
	m2 : CreateDeployment
	m2 : CreateEnvironment
	m2 : DeleteApplication
	m2 : DeleteApplicationFromEnvironment
	m2 : DeleteEnvironment
	m2 : GetApplication
	m2 : GetApplicationVersion
	m2 : GetBatchJobExecution
	m2 : GetDataSetDetails
	m2 : GetDataSetExportTask
	m2 : GetDataSetImportTask
	m2 : GetDeployment
	m2 : GetEnvironment
	m2 : GetSignedBluinsightsUrl
	m2 : ListApplicationVersions
	m2 : ListApplications
	m2 : ListBatchJobDefinitions

Préfixe de service	Actions
	m2 : ListBatchJobExecutions
	m2 : ListBatchJobRestartPoints
	m2 : ListDataSetExportHistory
	m2 : ListDataSetImportHistory
	m2 : ListDataSets
	m2 : ListDeployments
	m2 : ListEngineVersions
	m2 : ListEnvironments
	m2 : StartApplication
	m2 : StartBatchJob
	m2 : StopApplication
	m2 : UpdateApplication
	m2 : UpdateEnvironment

Préfixe de service	Actions
managedblockchain	blockchain gérée : CreateAccessor blockchain gérée : CreateMember blockchain gérée : CreateNetwork blockchain gérée : CreateNode blockchain gérée : CreateProposal blockchain gérée : DeleteAccessor blockchain gérée : DeleteMember blockchain gérée : DeleteNode blockchain gérée : GetAccessor blockchain gérée : GetMember blockchain gérée : GetNetwork blockchain gérée : GetNode blockchain gérée : GetProposal blockchain gérée : InvokeRpcPolygonMainnet blockchain gérée : InvokeRpcPolygonMumbaiTestnet blockchain gérée : ListAccessors blockchain gérée : ListInvitations blockchain gérée : ListMembers blockchain gérée : ListNetworks blockchain gérée : ListNodes blockchain gérée : ListProposalVotes

Préfixe de service	Actions
	blockchain gérée : ListProposals blockchain gérée : RejectInvitation blockchain gérée : UpdateMember blockchain gérée : UpdateNode blockchain gérée : VoteOnProposal

Préfixe de service	Actions
mediaconnect	connexion multimédia : AddBridgeOutputs
	connexion multimédia : AddBridgeSources
	connexion multimédia : AddFlowMediaStreams
	connexion multimédia : AddFlowOutputs
	connexion multimédia : AddFlowSources
	connexion multimédia : AddFlowVpcInterfaces
	connexion multimédia : CreateBridge
	connexion multimédia : CreateFlow
	connexion multimédia : CreateGateway
	connexion multimédia : DeleteBridge
	connexion multimédia : DeleteFlow
	connexion multimédia : DeleteGateway
	connexion multimédia : DeleteRouterInput
	connexion multimédia : DeleteRouterNetworkInterface
	connexion multimédia : DeleteRouterOutput
	connexion multimédia : DeregisterGatewayInstance
	connexion multimédia : DescribeBridge
	connexion multimédia : DescribeFlow
	connexion multimédia : DescribeFlowSourceMetadata
	connexion multimédia : DescribeFlowSourceThumbnail
	connexion multimédia : DescribeGateway

Préfixe de service	Actions
	connexion multimédia : DescribeGatewayInstance
	connexion multimédia : DescribeOffering
	connexion multimédia : DescribeReservation
	connexion multimédia : GetRouterInput
	connexion multimédia : GetRouterInputSourceMetadata
	connexion multimédia : GetRouterInputThumbnail
	connexion multimédia : GetRouterNetworkInterface
	connexion multimédia : GetRouterOutput
	connexion multimédia : GrantFlowEntitlements
	connexion multimédia : ListBridges
	connexion multimédia : ListEntitlements
	connexion multimédia : ListFlows
	connexion multimédia : ListGatewayInstances
	connexion multimédia : ListGateways
	connexion multimédia : ListOfferings
	connexion multimédia : ListReservations
	connexion multimédia : ListRouterInputs
	connexion multimédia : ListRouterNetworkInterfaces
	connexion multimédia : ListRouterOutputs
	connexion multimédia : PurchaseOffering
	connexion multimédia : RemoveBridgeOutput

Préfixe de service	Actions
	connexion multimédia : RemoveBridgeSource
	connexion multimédia : RemoveFlowMediaStream
	connexion multimédia : RemoveFlowOutput
	connexion multimédia : RemoveFlowSource
	connexion multimédia : RemoveFlowVpcInterface
	connexion multimédia : RestartRouterInput
	connexion multimédia : RestartRouterOutput
	connexion multimédia : RevokeFlowEntitlement
	connexion multimédia : StartFlow
	connexion multimédia : StartRouterInput
	connexion multimédia : StartRouterOutput
	connexion multimédia : StopFlow
	connexion multimédia : StopRouterInput
	connexion multimédia : StopRouterOutput
	connexion multimédia : TakeRouterInput
	connexion multimédia : UpdateBridge
	connexion multimédia : UpdateBridgeOutput
	connexion multimédia : UpdateBridgeSource
	connexion multimédia : UpdateBridgeState
	connexion multimédia : UpdateFlow
	connexion multimédia : UpdateFlowEntitlement

Préfixe de service	Actions
	connexion multimédia : UpdateFlowMediaStream connexion multimédia : UpdateGatewayInstance

Préfixe de service	Actions
mediaconvert	conversion de média : AssociateCertificate conversion de média : CancelJob conversion de média : CreateJob conversion de média : CreateJobTemplate conversion de média : CreatePreset conversion de média : CreateQueue conversion de média : CreateResourceShare conversion de média : DeleteJobTemplate conversion de média : DeletePolicy conversion de média : DeletePreset conversion de média : DeleteQueue conversion de média : DescribeEndpoints conversion de média : DisassociateCertificate conversion de média : GetJob conversion de média : GetJobTemplate conversion de média : GetPolicy conversion de média : GetPreset conversion de média : GetQueue conversion de média : ListJobTemplates conversion de média : ListJobs conversion de média : ListPresets

Préfixe de service	Actions
	conversion de média : ListQueues conversion de média : ListVersions mediaconvert:Probe conversion de média : PutPolicy conversion de média : SearchJobs conversion de média : UpdateJobTemplate conversion de média : UpdatePreset conversion de média : UpdateQueue

Préfixe de service	Actions
medialive	MediaLive : AcceptInputDeviceTransfer
	MediaLive : BatchDelete
	MediaLive : BatchStart
	MediaLive : BatchStop
	MediaLive : BatchUpdateSchedule
	MediaLive : CancelInputDeviceTransfer
	MediaLive : ClaimDevice
	MediaLive : CreateChannel
	MediaLive : CreateChannelPlacementGroup
	MediaLive : CreateCloudWatchAlarmTemplate
	MediaLive : CreateCloudWatchAlarmTemplateGroup
	MediaLive : CreateCluster
	MediaLive : CreateEventBridgeRuleTemplate
	MediaLive : CreateEventBridgeRuleTemplateGroup
	MediaLive : CreateInput
	MediaLive : CreateInputSecurityGroup
	MediaLive : CreateMultiplex
	MediaLive : CreateMultiplexProgram
	MediaLive : CreateNetwork
	MediaLive : CreateNode
	MediaLive : CreateNodeRegistrationScript

Préfixe de service	Actions
	MediaLive : CreatePartnerInput
	MediaLive : CreateSdiSource
	MediaLive : CreateSignalMap
	MediaLive : DeleteChannel
	MediaLive : DeleteChannelPlacementGroup
	MediaLive : DeleteCloudWatchAlarmTemplate
	MediaLive : DeleteCloudWatchAlarmTemplateGroup
	MediaLive : DeleteCluster
	MediaLive : DeleteEventBridgeRuleTemplate
	MediaLive : DeleteEventBridgeRuleTemplateGroup
	MediaLive : DeleteInput
	MediaLive : DeleteInputSecurityGroup
	MediaLive : DeleteMultiplex
	MediaLive : DeleteMultiplexProgram
	MediaLive : DeleteNetwork
	MediaLive : DeleteNode
	MediaLive : DeleteReservation
	MediaLive : DeleteSchedule
	MediaLive : DeleteSdiSource
	MediaLive : DeleteSignalMap
	MediaLive : DescribeAccountConfiguration

Préfixe de service	Actions
	MediaLive : DescribeChannel
	MediaLive : DescribeChannelPlacementGroup
	MediaLive : DescribeCluster
	MediaLive : DescribeInput
	MediaLive : DescribeInputDevice
	MediaLive : DescribeInputDeviceThumbnail
	MediaLive : DescribeInputSecurityGroup
	MediaLive : DescribeMultiplex
	MediaLive : DescribeMultiplexProgram
	MediaLive : DescribeNetwork
	MediaLive : DescribeNode
	MediaLive : DescribeOffering
	MediaLive : DescribeReservation
	MediaLive : DescribeSchedule
	MediaLive : DescribeSdiSource
	MediaLive : DescribeThumbnails
	MediaLive : GetCloudWatchAlarmTemplate
	MediaLive : GetCloudWatchAlarmTemplateGroup
	MediaLive : GetEventBridgeRuleTemplate
	MediaLive : GetEventBridgeRuleTemplateGroup
	MediaLive : GetSignalMap

Préfixe de service	Actions
	MediaLive : ListAlerts
	MediaLive : ListChannelPlacementGroups
	MediaLive : ListChannels
	MediaLive : ListCloudWatchAlarmTemplateGroups
	MediaLive : ListCloudWatchAlarmTemplates
	MediaLive : ListClusterAlerts
	MediaLive : ListClusters
	MediaLive : ListEventBridgeRuleTemplateGroups
	MediaLive : ListEventBridgeRuleTemplates
	MediaLive : ListInputDeviceTransfers
	MediaLive : ListInputDevices
	MediaLive : ListInputSecurityGroups
	MediaLive : ListInputs
	MediaLive : ListMultiplexAlerts
	MediaLive : ListMultiplexPrograms
	MediaLive : ListMultiplexes
	MediaLive : ListNetworks
	MediaLive : ListNodes
	MediaLive : ListOfferings
	MediaLive : ListReservations
	MediaLive : ListSdiSources

Préfixe de service	Actions
	MediaLive : ListSignalMaps
	MediaLive : ListVersions
	MediaLive : PurchaseOffering
	MediaLive : RebootInputDevice
	MediaLive : RejectInputDeviceTransfer
	MediaLive : RestartChannelPipelines
	MediaLive : StartChannel
	MediaLive : StartDeleteMonitorDeployment
	MediaLive : StartInputDevice
	MediaLive : StartInputDeviceMaintenanceWindow
	MediaLive : StartMonitorDeployment
	MediaLive : StartMultiplex
	MediaLive : StartUpdateSignalMap
	MediaLive : StopChannel
	MediaLive : StopInputDevice
	MediaLive : StopMultiplex
	MediaLive : TransferInputDevice
	MediaLive : UpdateAccountConfiguration
	MediaLive : UpdateChannel
	MediaLive : UpdateChannelClass
	MediaLive : UpdateChannelPlacementGroup

Préfixe de service	Actions
	MediaLive : UpdateCloudWatchAlarmTemplate
	MediaLive : UpdateCloudWatchAlarmTemplateGroup
	MediaLive : UpdateCluster
	MediaLive : UpdateEventBridgeRuleTemplate
	MediaLive : UpdateEventBridgeRuleTemplateGroup
	MediaLive : UpdateInput
	MediaLive : UpdateInputDevice
	MediaLive : UpdateInputSecurityGroup
	MediaLive : UpdateMultiplex
	MediaLive : UpdateMultiplexProgram
	MediaLive : UpdateNetwork
	MediaLive : UpdateNode
	MediaLive : UpdateNodeState
	MediaLive : UpdateReservation
	MediaLive : UpdateSdiSource

Préfixe de service	Actions
mediatailor	tailleur multimédia : ConfigureLogsForPlaybackConfiguration tailleur multimédia : CreateChannel tailleur multimédia : CreateLiveSource tailleur multimédia : CreatePrefetchSchedule tailleur multimédia : CreateProgram tailleur multimédia : CreateSourceLocation tailleur multimédia : CreateVodSource tailleur multimédia : DeleteChannel tailleur multimédia : DeleteChannelPolicy tailleur multimédia : DeleteLiveSource tailleur multimédia : DeletePlaybackConfiguration tailleur multimédia : DeletePrefetchSchedule tailleur multimédia : DeleteProgram tailleur multimédia : DeleteSourceLocation tailleur multimédia : DeleteVodSource tailleur multimédia : DescribeChannel tailleur multimédia : DescribeLiveSource tailleur multimédia : DescribeProgram tailleur multimédia : DescribeSourceLocation tailleur multimédia : DescribeVodSource tailleur multimédia : GetChannelPolicy

Préfixe de service	Actions
	tailleur multimédia : GetChannelSchedule tailleur multimédia : GetPlaybackConfiguration tailleur multimédia : GetPrefetchSchedule tailleur multimédia : ListAlerts tailleur multimédia : ListChannels tailleur multimédia : ListLiveSources tailleur multimédia : ListPlaybackConfigurations tailleur multimédia : ListPrefetchSchedules tailleur multimédia : ListSourceLocations tailleur multimédia : ListVodSources tailleur multimédia : PutChannelPolicy tailleur multimédia : PutPlaybackConfiguration tailleur multimédia : StartChannel tailleur multimédia : StopChannel tailleur multimédia : UpdateChannel tailleur multimédia : UpdateLiveSource tailleur multimédia : UpdateProgram tailleur multimédia : UpdateSourceLocation tailleur multimédia : UpdateVodSource

Préfixe de service	Actions
base de données de mémoire	base de données de mémoire : BatchUpdateCluster base de données de mémoire : CopySnapshot base de données de mémoire : CreateAcl base de données de mémoire : CreateCluster base de données de mémoire : CreateMultiRegionCluster base de données de mémoire : CreateParameterGroup base de données de mémoire : CreateSnapshot base de données de mémoire : CreateSubnetGroup base de données de mémoire : CreateUser base de données de mémoire : DeleteAcl base de données de mémoire : DeleteCluster base de données de mémoire : DeleteMultiRegionCluster base de données de mémoire : DeleteParameterGroup base de données de mémoire : DeleteSnapshot base de données de mémoire : DeleteSubnetGroup base de données de mémoire : DeleteUser base de données de mémoire : DescribeAcls base de données de mémoire : DescribeClusters base de données de mémoire : DescribeEngineVersions base de données de mémoire : DescribeEvents base de données de mémoire : DescribeMultiRegionClusters

Préfixe de service	Actions
	base de données de mémoire : DescribeMultiRegionParameterGroups
	base de données de mémoire : DescribeMultiRegionParameters
	base de données de mémoire : DescribeParameterGroups
	base de données de mémoire : DescribeParameters
	base de données de mémoire : DescribeReservedNodes
	base de données de mémoire : DescribeReservedNodesOfferings
	base de données de mémoire : DescribeServiceUpdates
	base de données de mémoire : DescribeSnapshots
	base de données de mémoire : DescribeSubnetGroups
	base de données de mémoire : DescribeUsers
	base de données de mémoire : FailoverShard
	base de données de mémoire : ListAllowedMultiRegionClusterUpdates
	base de données de mémoire : ListAllowedNodeTypeUpdates
	base de données de mémoire : PurchaseReservedNodesOffering
	base de données de mémoire : ResetParameterGroup
	base de données de mémoire : UpdateAcl
	base de données de mémoire : UpdateCluster
	base de données de mémoire : UpdateMultiRegionCluster
	base de données de mémoire : UpdateParameterGroup
	base de données de mémoire : UpdateSubnetGroup

Préfixe de service	Actions
	base de données de mémoire : UpdateUser

Préfixe de service	Actions
mgh	mgh : AssociateCreatedArtifact mgh : AssociateDiscoveredResource mgh : AssociateSourceResource mgh : CreateHomeRegionControl mgh : CreateProgressUpdateStream mgh : DeleteHomeRegionControl mgh : DeleteProgressUpdateStream mgh : DescribeApplicationState mgh : DescribeHomeRegionControls mgh : DescribeMigrationTask mgh : DisassociateCreatedArtifact mgh : DisassociateDiscoveredResource mgh : DisassociateSourceResource mgh : GetHomeRegion mgh : ImportMigrationTask mgh : ListApplicationStates mgh : ListCreatedArtifacts mgh : ListDiscoveredResources mgh : ListMigrationTaskUpdates mgh : ListMigrationTasks mgh : ListProgressUpdateStreams

Préfixe de service	Actions
	mgh : ListSourceResources mgh : NotifyApplicationState mgh : NotifyMigrationTaskState mgh : PutResourceAttributes

Préfixe de service	Actions
mgn	mitrailleuse : ArchiveApplication
	mitrailleuse : ArchiveWave
	mitrailleuse : AssociateApplications
	mitrailleuse : AssociateSourceServers
	mitrailleuse : ChangeServerLifeCycleState
	mitrailleuse : CreateApplication
	mitrailleuse : CreateConnector
	mitrailleuse : CreateLaunchConfigurationTemplate
	mitrailleuse : CreateNetworkMigrationDefinition
	mitrailleuse : CreateReplicationConfigurationTemplate
	mitrailleuse : CreateWave
	mitrailleuse : DeleteApplication
	mitrailleuse : DeleteConnector
	mitrailleuse : DeleteJob
	mitrailleuse : DeleteLaunchConfigurationTemplate
	mitrailleuse : DeleteNetworkMigrationDefinition
	mitrailleuse : DeleteReplicationConfigurationTemplate
	mitrailleuse : DeleteSourceServer
	mitrailleuse : DeleteVcenterClient
	mitrailleuse : DeleteWave
	mitrailleuse : DescribeJobLogItems

Préfixe de service	Actions
	mitrailleuse : DescribeJobs
	mitrailleuse : DescribeLaunchConfigurationTemplates
	mitrailleuse : DescribeReplicationConfigurationTemplates
	mitrailleuse : DescribeVcenterClients
	mitrailleuse : DisassociateApplications
	mitrailleuse : DisassociateSourceServers
	mitrailleuse : DisconnectFromService
	mitrailleuse : FinalizeCutover
	mitrailleuse : GetNetworkMigrationDefinition
	mitrailleuse : GetNetworkMigrationMapperSegmentConstruct
	mitrailleuse : GetReplicationConfiguration
	mitrailleuse : InitializeService
	mitrailleuse : ListConnectors
	mitrailleuse : ListExportErrors
	mitrailleuse : ListExports
	mitrailleuse : ListImportErrors
	mitrailleuse : ListImportFileEnrichments
	mitrailleuse : ListImports
	mitrailleuse : ListManagedAccounts
	mitrailleuse : ListNetworkMigrationAnalyses
	mitrailleuse : ListNetworkMigrationAnalysisResults

Préfixe de service	Actions
	mitrailleuse : ListNetworkMigrationCodeGenerationSegments
	mitrailleuse : ListNetworkMigrationCodeGenerations
	mitrailleuse : ListNetworkMigrationDefinitions
	mitrailleuse : ListNetworkMigrationDeployedStacks
	mitrailleuse : ListNetworkMigrationDeployments
	mitrailleuse : ListNetworkMigrationExecutions
	mitrailleuse : ListNetworkMigrationMapperSegmentConstructs
	mitrailleuse : ListNetworkMigrationMapperSegments
	mitrailleuse : ListNetworkMigrationMappingUpdates
	mitrailleuse : ListNetworkMigrationMappings
	mitrailleuse : ListSourceServerActions
	mitrailleuse : ListTemplateActions
	mitrailleuse : MarkAsArchived
	mitrailleuse : PauseReplication
	mitrailleuse : PutSourceServerAction
	mitrailleuse : PutTemplateAction
	mitrailleuse : RemoveSourceServerAction
	mitrailleuse : RemoveTemplateAction
	mitrailleuse : ResumeReplication
	mitrailleuse : RetryDataReplication
	mitrailleuse : StartCutover

Préfixe de service	Actions
	mitrailleuse : StartExport
	mitrailleuse : StartImport
	mitrailleuse : StartImportFileEnrichment
	mitrailleuse : StartNetworkMigrationCodeGeneration
	mitrailleuse : StartNetworkMigrationMapping
	mitrailleuse : StartNetworkMigrationMappingUpdate
	mitrailleuse : StartReplication
	mitrailleuse : StartTest
	mitrailleuse : StopReplication
	mitrailleuse : TerminateTargetInstances
	mitrailleuse : UnarchiveApplication
	mitrailleuse : UnarchiveWave
	mitrailleuse : UpdateApplication
	mitrailleuse : UpdateConnector
	mitrailleuse : UpdateLaunchConfigurationTemplate
	mitrailleuse : UpdateNetworkMigrationDefinition
	mitrailleuse : UpdateNetworkMigrationMapperSegment
	mitrailleuse : UpdateReplicationConfiguration
	mitrailleuse : UpdateReplicationConfigurationTemplate
	mitrailleuse : UpdateSourceServer
	mitrailleuse : UpdateSourceServerReplicationType

Préfixe de service	Actions
	mitrailleuse : UpdateWave

Préfixe de service	Actions
migrationhub-strategy	stratégie du pôle migratoire : GetAntiPattern stratégie du pôle migratoire : GetApplicationComponentDetails stratégie du pôle migratoire : GetApplicationComponentStrategies stratégie du pôle migratoire : GetAssessment stratégie du pôle migratoire : GetImportFileTask stratégie du pôle migratoire : GetLatestAssessmentId stratégie du pôle migratoire : GetMessage stratégie du pôle migratoire : GetPortfolioPreferences stratégie du pôle migratoire : GetPortfolioSummary stratégie du pôle migratoire : GetRecommendationReportDetails stratégie du pôle migratoire : GetServerDetails stratégie du pôle migratoire : GetServerStrategies stratégie du pôle migratoire : ListAnalyzableServers stratégie du pôle migratoire : ListAntiPatterns stratégie du pôle migratoire : ListApplicationComponents stratégie du pôle migratoire : ListCollectors stratégie du pôle migratoire : ListImportFileTask stratégie du pôle migratoire : ListJarArtifacts stratégie du pôle migratoire : ListServers stratégie du pôle migratoire : PutLogData stratégie du pôle migratoire : PutMetricData

Préfixe de service	Actions
	stratégie du pôle migratoire : PutPortfolioPreferences stratégie du pôle migratoire : RegisterCollector stratégie du pôle migratoire : SendMessage stratégie du pôle migratoire : StartAssessment stratégie du pôle migratoire : StartImportFileTask stratégie du pôle migratoire : StartRecommendationReportGeneration stratégie du pôle migratoire : StopAssessment stratégie du pôle migratoire : UpdateApplicationComponentConfig stratégie du pôle migratoire : UpdateCollectorConfiguration stratégie du pôle migratoire : UpdateServerConfig

Préfixe de service	Actions
mobiletargeting	ciblage mobile : CreateApp
	ciblage mobile : CreateCampaign
	ciblage mobile : CreateEmailTemplate
	ciblage mobile : CreateExportJob
	ciblage mobile : CreateImportJob
	ciblage mobile : CreateInAppTemplate
	ciblage mobile : CreateJourney
	ciblage mobile : CreatePushTemplate
	ciblage mobile : CreateRecommenderConfiguration
	ciblage mobile : CreateSegment
	ciblage mobile : CreateSmsTemplate
	ciblage mobile : CreateVoiceTemplate
	ciblage mobile : DeleteAdmChannel
	ciblage mobile : DeleteApnsChannel
	ciblage mobile : DeleteApnsSandboxChannel
	ciblage mobile : DeleteApnsVoipChannel
	ciblage mobile : DeleteApnsVoipSandboxChannel
	ciblage mobile : DeleteApp
	ciblage mobile : DeleteBaiduChannel
	ciblage mobile : DeleteCampaign
	ciblage mobile : DeleteEmailChannel

Préfixe de service	Actions
	ciblage mobile : DeleteEmailTemplate
	ciblage mobile : DeleteEndpoint
	ciblage mobile : DeleteEventStream
	ciblage mobile : DeleteGcmChannel
	ciblage mobile : DeleteInAppTemplate
	ciblage mobile : DeleteJourney
	ciblage mobile : DeletePushTemplate
	ciblage mobile : DeleteRecommenderConfiguration
	ciblage mobile : DeleteSegment
	ciblage mobile : DeleteSmsChannel
	ciblage mobile : DeleteSmsTemplate
	ciblage mobile : DeleteUserEndpoints
	ciblage mobile : DeleteVoiceChannel
	ciblage mobile : DeleteVoiceTemplate
	ciblage mobile : GetAdmChannel
	ciblage mobile : GetApnsChannel
	ciblage mobile : GetApnsSandboxChannel
	ciblage mobile : GetApnsVoipChannel
	ciblage mobile : GetApnsVoipSandboxChannel
	ciblage mobile : GetApp
	ciblage mobile : GetApplicationDateRangeKpi

Préfixe de service	Actions
	ciblage mobile : GetApplicationSettings
	ciblage mobile : GetApps
	ciblage mobile : GetBaiduChannel
	ciblage mobile : GetCampaign
	ciblage mobile : GetCampaignActivities
	ciblage mobile : GetCampaignDateRangeKpi
	ciblage mobile : GetCampaignVersion
	ciblage mobile : GetCampaignVersions
	ciblage mobile : GetCampaigns
	ciblage mobile : GetChannels
	ciblage mobile : GetEmailChannel
	ciblage mobile : GetEmailTemplate
	ciblage mobile : GetEndpoint
	ciblage mobile : GetEventStream
	ciblage mobile : GetExportJob
	ciblage mobile : GetExportJobs
	ciblage mobile : GetGcmChannel
	ciblage mobile : GetImportJob
	ciblage mobile : GetImportJobs
	ciblage mobile : GetInAppMessages
	ciblage mobile : GetInAppTemplate

Préfixe de service	Actions
	ciblage mobile : GetJourney
	ciblage mobile : GetJourneyDateRangeKpi
	ciblage mobile : GetJourneyExecutionActivityMetrics
	ciblage mobile : GetJourneyExecutionMetrics
	ciblage mobile : GetJourneyRunExecutionActivityMetrics
	ciblage mobile : GetJourneyRunExecutionMetrics
	ciblage mobile : GetJourneyRuns
	ciblage mobile : GetPushTemplate
	ciblage mobile : GetRecommenderConfiguration
	ciblage mobile : GetRecommenderConfigurations
	ciblage mobile : GetSegment
	ciblage mobile : GetSegmentExportJobs
	ciblage mobile : GetSegmentImportJobs
	ciblage mobile : GetSegmentVersion
	ciblage mobile : GetSegmentVersions
	ciblage mobile : GetSegments
	ciblage mobile : GetSmsChannel
	ciblage mobile : GetSmsTemplate
	ciblage mobile : GetUserEndpoints
	ciblage mobile : GetVoiceChannel
	ciblage mobile : GetVoiceTemplate

Préfixe de service	Actions
	ciblage mobile : ListJourneys
	ciblage mobile : ListTemplateVersions
	ciblage mobile : ListTemplates
	ciblage mobile : PhoneNumberValidate
	ciblage mobile : PutEventStream
	ciblage mobile : RemoveAttributes
	ciblage mobile : UpdateAdmChannel
	ciblage mobile : UpdateApnsChannel
	ciblage mobile : UpdateApnsSandboxChannel
	ciblage mobile : UpdateApnsVoipChannel
	ciblage mobile : UpdateApnsVoipSandboxChannel
	ciblage mobile : UpdateApplicationSettings
	ciblage mobile : UpdateBaiduChannel
	ciblage mobile : UpdateCampaign
	ciblage mobile : UpdateEmailChannel
	ciblage mobile : UpdateEmailTemplate
	ciblage mobile : UpdateEndpoint
	ciblage mobile : UpdateEndpointsBatch
	ciblage mobile : UpdateGcmChannel
	ciblage mobile : UpdateInAppTemplate
	ciblage mobile : UpdateJourney

Préfixe de service	Actions
	ciblage mobile : UpdateJourneyState
	ciblage mobile : UpdatePushTemplate
	ciblage mobile : UpdateRecommenderConfiguration
	ciblage mobile : UpdateSegment
	ciblage mobile : UpdateSmsChannel
	ciblage mobile : UpdateSmsTemplate
	ciblage mobile : UpdateTemplateActiveVersion
	ciblage mobile : UpdateVoiceChannel
	ciblage mobile : UpdateVoiceTemplate
	mobiletargeting:VerifyOTPMessage

Préfixe de service	Actions
mq	mq : CreateBroker mq : CreateConfiguration mq : CreateUser mq : DeleteBroker mq : DeleteConfiguration mq : DeleteUser mq : DescribeBroker mq : DescribeBrokerEngineTypes mq : DescribeBrokerInstanceOptions mq : DescribeConfiguration mq : DescribeConfigurationRevision mq : DescribeSharedResources mq : DescribeUser mq : ListBrokers mq : ListConfigurationRevisions mq : ListConfigurations mq : ListUsers mq:Promote mq : RebootBroker mq : UpdateBroker mq : UpdateConfiguration

Préfixe de service	Actions
	mq : UpdateUser

Préfixe de service	Actions
networkmanager	gestionnaire de réseau : AcceptAttachment
	gestionnaire de réseau : AssociateConnectPeer
	gestionnaire de réseau : AssociateCustomerGateway
	gestionnaire de réseau : AssociateLink
	gestionnaire de réseau : AssociateTransitGatewayConnectPeer
	gestionnaire de réseau : CreateConnectAttachment
	gestionnaire de réseau : CreateConnectPeer
	gestionnaire de réseau : CreateConnection
	gestionnaire de réseau : CreateCoreNetwork
	gestionnaire de réseau : CreateDevice
	gestionnaire de réseau : CreateDirectConnectGatewayAttachment
	gestionnaire de réseau : CreateGlobalNetwork
	gestionnaire de réseau : CreateLink
	gestionnaire de réseau : CreateSite
	gestionnaire de réseau : CreateSiteToSiteVpnAttachment
	gestionnaire de réseau : CreateTransitGatewayPeering
	gestionnaire de réseau : CreateTransitGatewayRouteTableAttachment
	gestionnaire de réseau : CreateVpcAttachment
	gestionnaire de réseau : DeleteAttachment
	gestionnaire de réseau : DeleteConnectPeer

Préfixe de service	Actions
	gestionnaire de réseau : DeleteConnection
	gestionnaire de réseau : DeleteCoreNetwork
	gestionnaire de réseau : DeleteCoreNetworkPolicyVersion
	gestionnaire de réseau : DeleteDevice
	gestionnaire de réseau : DeleteGlobalNetwork
	gestionnaire de réseau : DeleteLink
	gestionnaire de réseau : DeletePeering
	gestionnaire de réseau : DeleteResourcePolicy
	gestionnaire de réseau : DeleteSite
	gestionnaire de réseau : DeregisterTransitGateway
	gestionnaire de réseau : DescribeGlobalNetworks
	gestionnaire de réseau : DisassociateConnectPeer
	gestionnaire de réseau : DisassociateCustomerGateway
	gestionnaire de réseau : DisassociateLink
	gestionnaire de réseau : DisassociateTransitGatewayConnectPeer
	gestionnaire de réseau : ExecuteCoreNetworkChangeSet
	gestionnaire de réseau : GetConnectAttachment
	gestionnaire de réseau : GetConnectPeer
	gestionnaire de réseau : GetConnectPeerAssociations
	gestionnaire de réseau : GetConnections
	gestionnaire de réseau : GetCoreNetwork

Préfixe de service	Actions
	gestionnaire de réseau : GetCoreNetworkChangeEvents
	gestionnaire de réseau : GetCoreNetworkChangeSet
	gestionnaire de réseau : GetCoreNetworkPolicy
	gestionnaire de réseau : GetCustomerGatewayAssociations
	gestionnaire de réseau : GetDevices
	gestionnaire de réseau : GetLinkAssociations
	gestionnaire de réseau : GetLinks
	gestionnaire de réseau : GetNetworkResourceCounts
	gestionnaire de réseau : GetNetworkResourceRelationships
	gestionnaire de réseau : GetNetworkResources
	gestionnaire de réseau : GetNetworkRoutes
	gestionnaire de réseau : GetNetworkTelemetry
	gestionnaire de réseau : GetResourcePolicy
	gestionnaire de réseau : GetRouteAnalysis
	gestionnaire de réseau : GetSiteToSiteVpnAttachment
	gestionnaire de réseau : GetSites
	gestionnaire de réseau : GetTransitGatewayConnectPeerAssociations
	gestionnaire de réseau : GetTransitGatewayPeering
	gestionnaire de réseau : GetTransitGatewayRegistrations
	gestionnaire de réseau : GetTransitGatewayRouteTableAttachment

Préfixe de service	Actions
	gestionnaire de réseau : GetVpcAttachment
	gestionnaire de réseau : ListAttachmentRoutingPolicyAssociations
	gestionnaire de réseau : ListAttachments
	gestionnaire de réseau : ListConnectPeers
	gestionnaire de réseau : ListCoreNetworkPolicyVersions
	gestionnaire de réseau : ListCoreNetworkPrefixListAssociations
	gestionnaire de réseau : ListCoreNetworkRoutingInformation
	gestionnaire de réseau : ListCoreNetworks
	gestionnaire de réseau : ListOrganizationServiceAccessStatus
	gestionnaire de réseau : ListPeerings
	gestionnaire de réseau : PutAttachmentRoutingPolicyLabel
	gestionnaire de réseau : PutCoreNetworkPolicy
	gestionnaire de réseau : PutResourcePolicy
	gestionnaire de réseau : RegisterTransitGateway
	gestionnaire de réseau : RejectAttachment
	gestionnaire de réseau : RemoveAttachmentRoutingPolicyLabel
	gestionnaire de réseau : RestoreCoreNetworkPolicyVersion
	gestionnaire de réseau : StartOrganizationServiceAccessUpdate
	gestionnaire de réseau : StartRouteAnalysis
	gestionnaire de réseau : UpdateConnection
	gestionnaire de réseau : UpdateCoreNetwork

Préfixe de service	Actions
	gestionnaire de réseau : UpdateDevice
	gestionnaire de réseau : UpdateDirectConnectGatewayAttachment
	gestionnaire de réseau : UpdateGlobalNetwork
	gestionnaire de réseau : UpdateLink
	gestionnaire de réseau : UpdateNetworkResourceMetadata
	gestionnaire de réseau : UpdateSite
	gestionnaire de réseau : UpdateVpcAttachment

Préfixe de service	Actions
nimble	agile : AcceptEulas agile : CreateLaunchProfile agile : CreateStreamingImage agile : CreateStreamingSession agile : CreateStreamingSessionStream agile : CreateStudio agile : CreateStudioComponent agile : DeleteLaunchProfile agile : DeleteLaunchProfileMember agile : DeleteStreamingImage agile : DeleteStreamingSession agile : DeleteStudio agile : DeleteStudioComponent agile : DeleteStudioMember agile : GetEula agile : GetLaunchProfileDetails agile : GetStreamingImage agile : GetStreamingSession agile : GetStreamingSessionBackup agile : GetStreamingSessionStream agile : GetStudio

Préfixe de service	Actions
	agile : GetStudioComponent agile : GetStudioMember agile : ListEulas agile : ListLaunchProfileMembers agile : ListLaunchProfiles agile : ListStreamingImages agile : ListStreamingSessionBackups agile : ListStreamingSessions agile : ListStudioComponents agile : ListStudioMembers agile : ListStudios agile : PutLaunchProfileMembers agile : PutStudioMembers agile : StartStreamingSession agile : StartStudioSSOConfigurationRepair agile : StopStreamingSession agile : UpdateLaunchProfile agile : UpdateLaunchProfileMember agile : UpdateStreamingImage agile : UpdateStudio agile : UpdateStudioComponent

Préfixe de service	Actions
omics	bandes dessinées : AbortMultipartReadSetUpload
	bandes dessinées : AcceptShare
	bandes dessinées : BatchDeleteReadSet
	bandes dessinées : CancelAnnotationImportJob
	bandes dessinées : CancelRun
	bandes dessinées : CancelRunBatch
	bandes dessinées : CancelVariantImportJob
	bandes dessinées : CompleteMultipartReadSetUpload
	bandes dessinées : CreateAnnotationStore
	bandes dessinées : CreateAnnotationStoreVersion
	bandes dessinées : CreateConfiguration
	bandes dessinées : CreateMultipartReadSetUpload
	bandes dessinées : CreateReferenceStore
	bandes dessinées : CreateRunGroup
	bandes dessinées : CreateSequenceStore
	bandes dessinées : CreateShare
	bandes dessinées : CreateVariantStore
	bandes dessinées : CreateWorkflow
	bandes dessinées : CreateWorkflowVersion
	bandes dessinées : DeleteAnnotationStore
	bandes dessinées : DeleteAnnotationStoreVersions

Préfixe de service	Actions
	bandes dessinées : DeleteBatch
	bandes dessinées : DeleteConfiguration
	bandes dessinées : DeleteReference
	bandes dessinées : DeleteReferenceStore
	bandes dessinées : DeleteRun
	bandes dessinées : DeleteRunBatch
	bandes dessinées : DeleteRunGroup
	bandes dessinées : DeleteSequenceStore
	bandes dessinées : DeleteShare
	bandes dessinées : DeleteVariantStore
	bandes dessinées : DeleteWorkflow
	bandes dessinées : DeleteWorkflowVersion
	bandes dessinées : GetAnnotationImportJob
	bandes dessinées : GetAnnotationStore
	bandes dessinées : GetAnnotationStoreVersion
	bandes dessinées : GetBatch
	bandes dessinées : GetConfiguration
	bandes dessinées : GetReadSet
	bandes dessinées : GetReadSetActivationJob
	bandes dessinées : GetReadSetExportJob
	bandes dessinées : GetReadSetImportJob

Préfixe de service	Actions
	bandes dessinées : GetReadSetMetadata
	bandes dessinées : GetReference
	bandes dessinées : GetReferenceImportJob
	bandes dessinées : GetReferenceMetadata
	bandes dessinées : GetReferenceStore
	bandes dessinées : GetRun
	bandes dessinées : GetRunGroup
	bandes dessinées : GetRunTask
	bandes dessinées : GetSequenceStore
	bandes dessinées : GetShare
	bandes dessinées : GetVariantImportJob
	bandes dessinées : GetVariantStore
	bandes dessinées : GetWorkflow
	bandes dessinées : GetWorkflowVersion
	bandes dessinées : ListAnnotationImportJobs
	bandes dessinées : ListAnnotationStoreVersions
	bandes dessinées : ListAnnotationStores
	bandes dessinées : ListBatch
	bandes dessinées : ListConfigurations
	bandes dessinées : ListMultipartReadSetUploads
	bandes dessinées : ListReadSetActivationJobs

Préfixe de service	Actions
	bandes dessinées : ListReadSetExportJobs
	bandes dessinées : ListReadSetImportJobs
	bandes dessinées : ListReadSetUploadParts
	bandes dessinées : ListReadSets
	bandes dessinées : ListReferenceImportJobs
	bandes dessinées : ListReferenceStores
	bandes dessinées : ListReferences
	bandes dessinées : ListRunGroups
	bandes dessinées : ListRunTasks
	bandes dessinées : ListRuns
	bandes dessinées : ListRunsInBatch
	bandes dessinées : ListSequenceStores
	bandes dessinées : ListShares
	bandes dessinées : ListVariantImportJobs
	bandes dessinées : ListVariantStores
	bandes dessinées : ListWorkflowVersions
	bandes dessinées : ListWorkflows
	bandes dessinées : StartAnnotationImportJob
	bandes dessinées : StartReadSetActivationJob
	bandes dessinées : StartReadSetExportJob
	bandes dessinées : StartReadSetImportJob

Préfixe de service	Actions
	bandes dessinées : StartReferenceImportJob
	bandes dessinées : StartRun
	bandes dessinées : StartRunBatch
	bandes dessinées : StartVariantImportJob
	bandes dessinées : UpdateAnnotationStore
	bandes dessinées : UpdateAnnotationStoreVersion
	bandes dessinées : UpdateRunGroup
	bandes dessinées : UpdateVariantStore
	bandes dessinées : UpdateWorkflow
	bandes dessinées : UpdateWorkflowVersion
	bandes dessinées : UploadReadSetPart

Préfixe de service	Actions
opsworks	opsworks : AssignInstance opsworks : AssignVolume opsworks : AssociateElasticIp opsworks : AttachElasticLoadBalancer opsworks : CloneStack opsworks : CreateApp opsworks : CreateDeployment opsworks : CreateInstance opsworks : CreateLayer opsworks : CreateStack opsworks : CreateUserProfile opsworks : DeleteApp opsworks : DeleteInstance opsworks : DeleteLayer opsworks : DeleteStack opsworks : DeleteUserProfile opsworks : DeregisterEcsCluster opsworks : DeregisterElasticIp opsworks : DeregisterInstance opsworks : DeregisterRdsDbInstance opsworks : DeregisterVolume

Préfixe de service	Actions
	opsworks : DescribeAgentVersions opsworks : DescribeApps opsworks : DescribeCommands opsworks : DescribeDeployments opsworks : DescribeEcsClusters opsworks : DescribeElasticIps opsworks : DescribeElasticLoadBalancers opsworks : DescribeInstances opsworks : DescribeLayers opsworks : DescribeLoadBasedAutoScaling opsworks : DescribeMyUserProfile opsworks : DescribeOperatingSystems opsworks : DescribePermissions opsworks : DescribeRaidArrays opsworks : DescribeRdsDbInstances opsworks : DescribeServiceErrors opsworks : DescribeStackProvisioningParameters opsworks : DescribeStackSummary opsworks : DescribeStacks opsworks : DescribeTimeBasedAutoScaling opsworks : DescribeUserProfiles

Préfixe de service	Actions
	opsworks : DescribeVolumes opsworks : DetachElasticLoadBalancer opsworks : DisassociateElasticIp opsworks : GetHostnameSuggestion opsworks : GrantAccess opsworks : RebootInstance opsworks : RegisterEcsCluster opsworks : RegisterElasticIp opsworks : RegisterInstance opsworks : RegisterRdsDbInstance opsworks : RegisterVolume opsworks : SetLoadBasedAutoScaling opsworks : SetPermission opsworks : SetTimeBasedAutoScaling opsworks : StartInstance opsworks : StartStack opsworks : StopInstance opsworks : StopStack opsworks : UnassignInstance opsworks : UnassignVolume opsworks : UpdateApp

Préfixe de service	Actions
	opsworks : UpdateElasticIp opsworks : UpdateInstance opsworks : UpdateLayer opsworks : UpdateMyUserProfile opsworks : UpdateRdsDbInstance opsworks : UpdateStack opsworks : UpdateUserProfile opsworks : UpdateVolume

Préfixe de service	Actions
opsworks-cm	Opsworks-cm : AssociateNode
	Opsworks-cm : CreateBackup
	Opsworks-cm : CreateServer
	Opsworks-cm : DeleteBackup
	Opsworks-cm : DeleteServer
	Opsworks-cm : DescribeAccountAttributes
	Opsworks-cm : DescribeBackups
	Opsworks-cm : DescribeEvents
	Opsworks-cm : DescribeNodeAssociationStatus
	Opsworks-cm : DescribeServers
	Opsworks-cm : DisassociateNode
	Opsworks-cm : ExportServerEngineAttribute
	Opsworks-cm : RestoreServer
	Opsworks-cm : StartMaintenance
	Opsworks-cm : UpdateServer
	Opsworks-cm : UpdateServerEngineAttributes

Préfixe de service	Actions
organisations	organisations : AcceptHandshake organisations : AttachPolicy organisations : CancelHandshake organisations : CloseAccount organisations : CreateAccount organisations : CreateGovCloudAccount organisations : CreateOrganization organisations : CreateOrganizationalUnit organisations : CreatePolicy organisations : DeclineHandshake organisations : DeleteOrganization organisations : DeleteOrganizationalUnit organisations : DeletePolicy organisations : DeleteResourcePolicy organisations : DeregisterDelegatedAdministrator organisations : DescribeAccount organisations : DescribeCreateAccountStatus organisations : DescribeEffectivePolicy organisations : DescribeHandshake organisations : DescribeOrganization organisations : DescribeOrganizationalUnit

Préfixe de service	Actions
	organisations : DescribePolicy organisations : DescribeResourcePolicy organisations : DescribeResponsibilityTransfer organisations : DetachPolicy organisations : DisableAWSServiceAccess organisations : DisablePolicyType organisations : EnableAWSServiceAccess organisations : EnableAllFeatures organisations : EnablePolicyType organisations : InviteAccountToOrganization organisations : LeaveOrganization organisations : ListAWSServiceAccessForOrganization organisations : ListAccounts organisations : ListAccountsForParent organisations : ListAccountsWithInvalidEffectivePolicy organisations : ListChildren organisations : ListCreateAccountStatus organisations : ListDelegatedAdministrators organisations : ListDelegatedServicesForAccount organisations : ListHandshakesForAccount organisations : ListHandshakesForOrganization

Préfixe de service	Actions
	organisations : ListInboundResponsibilityTransfers organisations : ListOrganizationalUnitsForParent organisations : ListOutboundResponsibilityTransfers organisations : ListParents organisations : ListPolicies organisations : ListPoliciesForTarget organisations : ListRoots organisations : ListTargetsForPolicy organisations : MoveAccount organisations : PutResourcePolicy organisations : RegisterDelegatedAdministrator organisations : RemoveAccountFromOrganization organisations : TerminateResponsibilityTransfer organisations : UpdateOrganizationalUnit organisations : UpdatePolicy organisations : UpdateResponsibilityTransfer

Préfixe de service	Actions
outposts	avant-postes : CancelCapacityTask
	avant-postes : CancelOrder
	avant-postes : CreateOrder
	avant-postes : CreateOutpost
	avant-postes : CreatePrivateConnectivityConfig
	avant-postes : CreateQuote
	avant-postes : CreateRenewal
	avant-postes : CreateSite
	avant-postes : DeleteOutpost
	avant-postes : DeleteQuote
	avant-postes : DeleteSite
	avant-postes : GetCapacityTask
	avant-postes : GetCatalogItem
	avant-postes : GetConnection
	avant-postes : GetOrder
	avant-postes : GetOutpost
	avant-postes : GetOutpostBillingInformation
	avant-postes : GetOutpostInstanceTypes
	avant-postes : GetOutpostSupportedInstanceTypes
	avant-postes : GetPrivateConnectivityConfig
	avant-postes : GetQuote

Préfixe de service	Actions
	avant-postes : GetRenewalPricing
	avant-postes : GetSite
	avant-postes : GetSiteAddress
	avant-postes : ListAssetInstances
	avant-postes : ListAssets
	avant-postes : ListBlockingInstancesForCapacityTask
	avant-postes : ListCapacityTasks
	avant-postes : ListCatalogItems
	avant-postes : ListOrderableInstanceTypes
	avant-postes : ListOrders
	avant-postes : ListOutposts
	avant-postes : ListQuotes
	avant-postes : ListSites
	avant-postes : StartCapacityTask
	avant-postes : StartConnection
	avant-postes : UpdateOutpost
	avant-postes : UpdateQuote
	avant-postes : UpdateSite
	avant-postes : UpdateSiteAddress
	avant-postes : UpdateSiteRackPhysicalProperties

Préfixe de service	Actions
pi	épi : CreatePerformanceAnalysisReport épi : DeletePerformanceAnalysisReport épi : DescribeDimensionKeys épi : GetDimensionKeyDetails épi : GetPerformanceAnalysisReport épi : GetResourceMetadata épi : GetResourceMetrics épi : ListAvailableResourceDimensions épi : ListAvailableResourceMetrics épi : ListPerformanceAnalysisReports
pipes	tuyaux : CreatePipe tuyaux : DeletePipe tuyaux : DescribePipe tuyaux : ListPipes tuyaux : StartPipe tuyaux : StopPipe tuyaux : UpdatePipe

Préfixe de service	Actions
polly	Polly : DeleteLexicon Polly : DescribeVoices Polly : GetLexicon Polly : GetSpeechSynthesisTask Polly : ListLexicons Polly : ListSpeechSynthesisTasks Polly : PutLexicon Polly : StartSpeechSynthesisTask Polly : SynthesizeSpeech

Préfixe de service	Actions
profile	profil : AddProfileKey
	profil : BatchGetCalculatedAttributeForProfile
	profil : BatchGetProfile
	profil : CreateCalculatedAttributeDefinition
	profil : CreateDomain
	profil : CreateEventStream
	profil : CreateProfile
	profil : CreateRecommender
	profil : CreateRecommenderFilter
	profil : CreateRecommenderSchema
	profil : CreateSegmentDefinition
	profil : CreateSegmentEstimate
	profil : CreateSegmentSnapshot
	profil : CreateUploadJob
	profil : DeleteCalculatedAttributeDefinition
	profil : DeleteDomain
	profil : DeleteDomainObjectType
	profil : DeleteEventStream
	profil : DeleteIntegration
	profil : DeleteProfile
	profil : DeleteProfileKey

Préfixe de service	Actions
	profil : DeleteProfileObject
	profil : DeleteProfileObjectType
	profil : DeleteRecommender
	profil : DeleteRecommenderFilter
	profil : DeleteRecommenderSchema
	profil : DeleteSegmentDefinition
	profil : DeleteWorkflow
	profil : DetectProfileObjectType
	profil : GetAutoMergingPreview
	profil : GetCalculatedAttributeDefinition
	profil : GetCalculatedAttributeForProfile
	profil : GetDomain
	profil : GetDomainObjectType
	profil : GetEventStream
	profil : GetIdentityResolutionJob
	profil : GetIntegration
	profil : GetMatches
	profil : GetObjectTypeAttributeStatistics
	profil : GetProfileObjectType
	profil : GetProfileObjectTypeTemplate
	profil : GetProfileRecommendations

Préfixe de service	Actions
	profil : GetRecommender
	profil : GetRecommenderFilter
	profil : GetRecommenderSchema
	profil : GetSegmentDefinition
	profil : GetSegmentEstimate
	profil : GetSegmentMembership
	profil : GetSegmentSnapshot
	profil : GetSimilarProfiles
	profil : GetUploadJob
	profil : GetUploadJobPath
	profil : GetWorkflow
	profil : GetWorkflowSteps
	profil : ListAccountIntegrations
	profil : ListCalculatedAttributeDefinitions
	profil : ListCalculatedAttributesForProfile
	profil : ListDomainLayouts
	profil : ListDomainObjectTypes
	profil : ListDomains
	profil : ListEventStreams
	profil : ListIdentityResolutionJobs
	profil : ListIntegrations

Préfixe de service	Actions
	profil : ListObjectTypeAttributeValues
	profil : ListObjectTypeAttributes
	profil : ListProfileAttributeValues
	profil : ListProfileObjectTypeTemplates
	profil : ListProfileObjectTypes
	profil : ListProfileObjects
	profil : ListRecommenderFilters
	profil : ListRecommenderRecipes
	profil : ListRecommenderSchemas
	profil : ListRecommenders
	profil : ListRuleBasedMatches
	profil : ListSegmentDefinitions
	profil : ListUploadJobs
	profil : ListWorkflows
	profil : MergeProfiles
	profil : PutDomainObjectType
	profil : PutIntegration
	profil : PutProfileObject
	profil : PutProfileObjectType
	profil : SearchProfiles
	profil : StartRecommender

Préfixe de service	Actions
	profil : StartUploadJob
	profil : StopRecommender
	profil : StopUploadJob
	profil : UpdateCalculatedAttributeDefinition
	profil : UpdateDomain
	profil : UpdateProfile
	profil : UpdateRecommender

Préfixe de service	Actions
qldb	qldb : CancelJournalKinesisStream qldb : CreateLedger qldb : DeleteLedger qldb : DescribeJournalKinesisStream qldb : DescribeJournalS3Export qldb : DescribeLedger qldb : ExportJournalToS3 qldb : GetBlock qldb : GetDigest qldb : GetRevision qldb : ListJournalKinesisStreamsForLedger qldb : ListJournalS3Exports qldb : ListJournalS3ExportsForLedger qldb : ListLedgers qldb : StreamJournalToKinesis qldb : UpdateLedger qldb : UpdateLedgerPermissionsMode

Préfixe de service	Actions
ram	RAM : AcceptResourceShareInvitation
	RAM : AssociateResourceShare
	RAM : AssociateResourceSharePermission
	RAM : CreatePermission
	RAM : CreatePermissionVersion
	RAM : CreateResourceShare
	RAM : DeletePermission
	RAM : DeletePermissionVersion
	RAM : DeleteResourceShare
	RAM : DisassociateResourceShare
	RAM : DisassociateResourceSharePermission
	RAM : EnableSharingWithAwsOrganization
	RAM : GetPermission
	RAM : GetResourcePolicies
	RAM : GetResourceShareAssociations
	RAM : GetResourceShareInvitations
	RAM : GetResourceShares
	RAM : ListPendingInvitationResources
	RAM : ListPermissionAssociations
	RAM : ListPermissionVersions
	RAM : ListPermissions

Préfixe de service	Actions
	RAM : ListPrincipals RAM : ListReplacePermissionAssociationsWork RAM : ListResourceSharePermissions RAM : ListResourceTypes RAM : ListResources RAM : ListSourceAssociations RAM : PromotePermissionCreatedFromPolicy RAM : PromoteResourceShareCreatedFromPolicy RAM : RejectResourceShareInvitation RAM : ReplacePermissionAssociations RAM : SetDefaultPermissionVersion RAM : UpdateResourceShare
rbin	corbeille : CreateRule corbeille : DeleteRule corbeille : GetRule corbeille : ListRules corbeille : LockRule corbeille : UnlockRule corbeille : UpdateRule

Préfixe de service	Actions
rds	rouge : AddRoleToDBCluster rouge : AddRoleToDBInstance rouge : AddSourceIdentifierToSubscription rouge : ApplyPendingMaintenanceAction rouge : AuthorizeDBSecurityGroupIngress rds:BacktrackDBCluster rouge : CancelExportTask rouge : CopyDBClusterParameterGroup rouge : CopyDBClusterSnapshot rouge : CopyDBParameterGroup rds:CopyDBSnapshot rouge : CopyOptionGroup rouge : CreateCustomDBEngineVersion rouge : CreateDBClusterParameterGroup rouge : CreateDBParameterGroup rds:CreateDBProxy rouge : CreateDBProxyEndpoint rouge : CreateDBSecurityGroup rouge : CreateDBSubnetGroup rouge : CreateEventSubscription rouge : CreateGlobalCluster

Préfixe de service	Actions
	rouge : CreateOptionGroup
	rouge : DeleteBlueGreenDeployment
	rouge : DeleteDBClusterAutomatedBackup
	rouge : DeleteDBClusterParameterGroup
	rouge : DeleteDBClusterSnapshot
	rouge : DeleteDBInstanceAutomatedBackup
	rouge : DeleteDBParameterGroup
	rds:DeleteDBProxy
	rouge : DeleteDBProxyEndpoint
	rouge : DeleteDBSecurityGroup
	rds:DeleteDBSnapshot
	rouge : DeleteDBSubnetGroup
	rouge : DeleteEventSubscription
	rouge : DeleteGlobalCluster
	rouge : DeleteOptionGroup
	rouge : DeregisterDBProxyTargets
	rouge : DescribeAccountAttributes
	rouge : DescribeBlueGreenDeployments
	rouge : DescribeCertificates
	rouge : DescribeDBClusterAutomatedBackups
	rouge : DescribeDBClusterBacktracks

Préfixe de service	Actions
	rouge : DescribeDBClusterEndpoints
	rouge : DescribeDBClusterParameterGroups
	rouge : DescribeDBClusterParameters
	rouge : DescribeDBClusterSnapshotAttributes
	rouge : DescribeDBClusterSnapshots
	rds:DescribeDBClusters
	rouge : DescribeDBEngineVersions
	rouge : DescribeDBInstanceAutomatedBackups
	rds:DescribeDBInstances
	rouge : DescribeDBLogFiles
	rouge : DescribeDBMajorEngineVersions
	rouge : DescribeDBParameterGroups
	rds:DescribeDBParameters
	rds:DescribeDBProxies
	rouge : DescribeDBProxyEndpoints
	rouge : DescribeDBProxyTargetGroups
	rouge : DescribeDBProxyTargets
	rds:DescribeDBRecommendations
	rouge : DescribeDBSecurityGroups
	rouge : DescribeDBSnapshotAttributes
	rouge : DescribeDBSnapshotTenantDatabases

Préfixe de service	Actions
	rds:DescribeDBSnapshots rouge : DescribeDBSubnetGroups rouge : DescribeEngineDefaultClusterParameters rouge : DescribeEngineDefaultParameters rouge : DescribeEventCategories rouge : DescribeEventSubscriptions rouge : DescribeEvents rouge : DescribeExportTasks rouge : DescribeGlobalClusters rouge : DescribeIntegrations rouge : DescribeOptionGroupOptions rouge : DescribeOptionGroups rouge : DescribeOrderableDBInstanceOptions rouge : DescribePendingMaintenanceActions rouge : DescribeReservedDBInstances rouge : DescribeReservedDBInstancesOfferings rouge : DescribeSourceRegions rouge : DescribeTenantDatabases rouge : DescribeValidDBInstanceModifications rouge : DownloadCompleteDBLogFile rouge : DownloadDBLogFilePortion

Préfixe de service	Actions
	rds:FailoverDBCluster
	rouge : FailoverGlobalCluster
	rouge : ModifyActivityStream
	rouge : ModifyCertificates
	rouge : ModifyCurrentDBClusterCapacity
	rouge : ModifyDBClusterEndpoint
	rouge : ModifyDBClusterParameterGroup
	rouge : ModifyDBClusterSnapshotAttribute
	rouge : ModifyDBParameterGroup
	rds:ModifyDBProxy
	rouge : ModifyDBProxyEndpoint
	rouge : ModifyDBProxyTargetGroup
	rds:ModifyDBRecommendation
	rds:ModifyDBSnapshot
	rouge : ModifyDBSnapshotAttribute
	rouge : ModifyDBSubnetGroup
	rouge : ModifyEventSubscription
	rouge : ModifyGlobalCluster
	rouge : ModifyOptionGroup
	rouge : ModifyTenantDatabase
	rouge : PurchaseReservedDBInstancesOffering

Préfixe de service	Actions
	rds:RebootDBCluster
	rouge : RegisterDBProxyTargets
	rouge : RemoveFromGlobalCluster
	rouge : RemoveRoleFromDBCluster
	rouge : RemoveRoleFromDBInstance
	rouge : RemoveSourceIdentifierFromSubscription
	rouge : ResetDBClusterParameterGroup
	rouge : ResetDBParameterGroup
	rouge : RestoreDBClusterFromS3
	rouge : RestoreDBClusterFromSnapshot
	rouge : RestoreDBClusterToPointInTime
	rouge : RestoreDBInstanceFromDBSnapshot
	rouge : RestoreDBInstanceFromS3
	rouge : RestoreDBInstanceToPointInTime
	rouge : RevokeDBSecurityGroupIngress
	rouge : StartActivityStream
	rds:StartDBCluster
	rds:StartDBInstance
	rouge : StartDBInstanceAutomatedBackupsReplication
	rouge : StartExportTask
	rouge : StopActivityStream

Préfixe de service	Actions
	rds:StopDBCluster rds:StopDBInstance rouge : StopDBInstanceAutomatedBackupsReplication rouge : SwitchoverBlueGreenDeployment rouge : SwitchoverGlobalCluster rouge : SwitchoverReadReplica

Préfixe de service	Actions
redshift	décalage vers le rouge : AcceptReservedNodeExchange décalage vers le rouge : AddPartner décalage vers le rouge : AssociateDataShareConsumer décalage vers le rouge : AuthorizeClusterSecurityGroupIngress décalage vers le rouge : AuthorizeDataShare décalage vers le rouge : AuthorizeEndpointAccess décalage vers le rouge : AuthorizeSnapshotAccess décalage vers le rouge : BatchDeleteClusterSnapshots décalage vers le rouge : BatchModifyClusterSnapshots décalage vers le rouge : CancelResize décalage vers le rouge : CopyClusterSnapshot décalage vers le rouge : CreateAuthenticationProfile décalage vers le rouge : CreateCluster décalage vers le rouge : CreateClusterParameterGroup décalage vers le rouge : CreateClusterSecurityGroup décalage vers le rouge : CreateClusterSnapshot décalage vers le rouge : CreateClusterSubnetGroup décalage vers le rouge : CreateCustomDomainAssociation décalage vers le rouge : CreateEndpointAccess décalage vers le rouge : CreateEventSubscription décalage vers le rouge : CreateHsmClientCertificate

Préfixe de service	Actions
	décalage vers le rouge : CreateHsmConfiguration décalage vers le rouge : CreateIntegration décalage vers le rouge : CreateRedshiftIdcApplication décalage vers le rouge : CreateScheduledAction décalage vers le rouge : CreateSnapshotCopyGrant décalage vers le rouge : CreateSnapshotSchedule décalage vers le rouge : CreateUsageLimit décalage vers le rouge : DeauthorizeDataShare décalage vers le rouge : DeleteAuthenticationProfile décalage vers le rouge : DeleteCluster décalage vers le rouge : DeleteClusterParameterGroup décalage vers le rouge : DeleteClusterSecurityGroup décalage vers le rouge : DeleteClusterSnapshot décalage vers le rouge : DeleteClusterSubnetGroup décalage vers le rouge : DeleteCustomDomainAssociation décalage vers le rouge : DeleteEndpointAccess décalage vers le rouge : DeleteEventSubscription décalage vers le rouge : DeleteHsmClientCertificate décalage vers le rouge : DeleteHsmConfiguration décalage vers le rouge : DeletePartner décalage vers le rouge : DeleteRedshiftIdcApplication

Préfixe de service	Actions
	décalage vers le rouge : DeleteResourcePolicy décalage vers le rouge : DeleteScheduledAction décalage vers le rouge : DeleteSnapshotCopyGrant décalage vers le rouge : DeleteSnapshotSchedule décalage vers le rouge : DeleteUsageLimit décalage vers le rouge : DeregisterNamespace décalage vers le rouge : DescribeAccountAttributes décalage vers le rouge : DescribeAuthenticationProfiles décalage vers le rouge : DescribeClusterDbRevisions décalage vers le rouge : DescribeClusterParameterGroups décalage vers le rouge : DescribeClusterParameters décalage vers le rouge : DescribeClusterSecurityGroups décalage vers le rouge : DescribeClusterSnapshots décalage vers le rouge : DescribeClusterSubnetGroups décalage vers le rouge : DescribeClusterTracks décalage vers le rouge : DescribeClusterVersions décalage vers le rouge : DescribeClusters décalage vers le rouge : DescribeCustomDomainAssociations décalage vers le rouge : DescribeDataShares décalage vers le rouge : DescribeDataSharesForConsumer décalage vers le rouge : DescribeDataSharesForProducer

Préfixe de service	Actions
	décalage vers le rouge : DescribeDefaultClusterParameters décalage vers le rouge : DescribeEndpointAccess décalage vers le rouge : DescribeEndpointAuthorization décalage vers le rouge : DescribeEventCategories décalage vers le rouge : DescribeEventSubscriptions décalage vers le rouge : DescribeEvents décalage vers le rouge : DescribeHsmClientCertificates décalage vers le rouge : DescribeHsmConfigurations décalage vers le rouge : DescribeInboundIntegrations décalage vers le rouge : DescribeIntegrations décalage vers le rouge : DescribeLoggingStatus décalage vers le rouge : DescribeNodeConfigurationOptions décalage vers le rouge : DescribeOrderableClusterOptions décalage vers le rouge : DescribePartners décalage vers le rouge : DescribeRedshiftIdcApplications décalage vers le rouge : DescribeReservedNodeExchangeStatus décalage vers le rouge : DescribeReservedNodeOfferings décalage vers le rouge : DescribeReservedNodes décalage vers le rouge : DescribeResize décalage vers le rouge : DescribeScheduledActions décalage vers le rouge : DescribeSnapshotCopyGrants

Préfixe de service	Actions
	décalage vers le rouge : DescribeSnapshotSchedules décalage vers le rouge : DescribeStorage décalage vers le rouge : DescribeTableRestoreStatus décalage vers le rouge : DescribeUsageLimits décalage vers le rouge : DisableLogging décalage vers le rouge : DisableSnapshotCopy décalage vers le rouge : DisassociateDataShareConsumer décalage vers le rouge : EnableLogging décalage vers le rouge : EnableSnapshotCopy décalage vers le rouge : FailoverPrimaryCompute décalage vers le rouge : GetClusterCredentials décalage vers le rouge : GetClusterCredentialsWithIAM décalage vers le rouge : GetIdentityCenterAuthToken décalage vers le rouge : GetReservedNodeExchangeConfiguration Options décalage vers le rouge : GetReservedNodeExchangeOfferings décalage vers le rouge : GetResourcePolicy décalage vers le rouge : ListRecommendations décalage vers le rouge : ModifyAquaConfiguration décalage vers le rouge : ModifyAuthenticationProfile décalage vers le rouge : ModifyCluster

Préfixe de service	Actions
	décalage vers le rouge : ModifyClusterDbRevision décalage vers le rouge : ModifyClusterIamRoles décalage vers le rouge : ModifyClusterMaintenance décalage vers le rouge : ModifyClusterParameterGroup décalage vers le rouge : ModifyClusterSnapshot décalage vers le rouge : ModifyClusterSnapshotSchedule décalage vers le rouge : ModifyClusterSubnetGroup décalage vers le rouge : ModifyCustomDomainAssociation décalage vers le rouge : ModifyEndpointAccess décalage vers le rouge : ModifyEventSubscription décalage vers le rouge : ModifyRedshiftIdcApplication décalage vers le rouge : ModifyScheduledAction décalage vers le rouge : ModifySnapshotCopyRetentionPeriod décalage vers le rouge : ModifySnapshotSchedule décalage vers le rouge : ModifyUsageLimit décalage vers le rouge : PauseCluster décalage vers le rouge : PurchaseReservedNodeOffering décalage vers le rouge : PutResourcePolicy décalage vers le rouge : RebootCluster décalage vers le rouge : RegisterNamespace décalage vers le rouge : RejectDataShare

Préfixe de service	Actions
	décalage vers le rouge : ResetClusterParameterGroup
	décalage vers le rouge : ResizeCluster
	décalage vers le rouge : RestoreFromClusterSnapshot
	décalage vers le rouge : RestoreTableFromClusterSnapshot
	décalage vers le rouge : ResumeCluster
	décalage vers le rouge : RevokeClusterSecurityGroupIngress
	décalage vers le rouge : RevokeEndpointAccess
	décalage vers le rouge : RevokeSnapshotAccess
	décalage vers le rouge : RotateEncryptionKey
	décalage vers le rouge : UpdatePartnerStatus
redshift-data	données redshift : BatchExecuteStatement
	données redshift : CancelStatement
	données redshift : DescribeStatement
	données redshift : DescribeTable
	données redshift : ExecuteStatement
	données redshift : GetStatementResult
	données redshift : ListDatabases
	données redshift : ListSchemas
	données redshift : ListSessions
	données redshift : ListStatements
	données redshift : ListTables

Préfixe de service	Actions
refactor-spaces	espaces de refactorisation : CreateApplication espaces de refactorisation : CreateEnvironment espaces de refactorisation : CreateRoute espaces de refactorisation : CreateService espaces de refactorisation : DeleteApplication espaces de refactorisation : DeleteEnvironment espaces de refactorisation : DeleteResourcePolicy espaces de refactorisation : DeleteRoute espaces de refactorisation : DeleteService espaces de refactorisation : GetApplication espaces de refactorisation : GetEnvironment espaces de refactorisation : GetResourcePolicy espaces de refactorisation : GetRoute espaces de refactorisation : GetService espaces de refactorisation : ListApplications espaces de refactorisation : ListEnvironmentVpcs espaces de refactorisation : ListEnvironments espaces de refactorisation : ListRoutes espaces de refactorisation : ListServices espaces de refactorisation : PutResourcePolicy espaces de refactorisation : UpdateRoute

Préfixe de service	Actions
rekognition	reconnaissance : AssociateFaces
	reconnaissance : CompareFaces
	reconnaissance : CopyProjectVersion
	reconnaissance : CreateCollection
	reconnaissance : CreateDataset
	reconnaissance : CreateFaceLivenessSession
	reconnaissance : CreateProject
	reconnaissance : CreateProjectVersion
	reconnaissance : CreateStreamProcessor
	reconnaissance : CreateUser
	reconnaissance : DeleteCollection
	reconnaissance : DeleteDataset
	reconnaissance : DeleteFaces
	reconnaissance : DeleteProject
	reconnaissance : DeleteProjectPolicy
	reconnaissance : DeleteProjectVersion
	reconnaissance : DeleteStreamProcessor
	reconnaissance : DeleteUser
	reconnaissance : DescribeCollection
	reconnaissance : DescribeDataset
	reconnaissance : DescribeProjectVersions

Préfixe de service	Actions
	reconnaissance : DescribeProjects
	reconnaissance : DescribeStreamProcessor
	reconnaissance : DetectCustomLabels
	reconnaissance : DetectFaces
	reconnaissance : DetectLabels
	reconnaissance : DetectModerationLabels
	reconnaissance : DetectProtectiveEquipment
	reconnaissance : DetectText
	reconnaissance : DisassociateFaces
	reconnaissance : DistributeDatasetEntries
	reconnaissance : GetCelebrityInfo
	reconnaissance : GetCelebrityRecognition
	reconnaissance : GetContentModeration
	reconnaissance : GetFaceDetection
	reconnaissance : GetFaceLivenessSessionResults
	reconnaissance : GetFaceSearch
	reconnaissance : GetLabelDetection
	reconnaissance : GetMediaAnalysisJob
	reconnaissance : GetPersonTracking
	reconnaissance : GetSegmentDetection
	reconnaissance : GetTextDetection

Préfixe de service	Actions
	reconnaissance : IndexFaces
	reconnaissance : ListCollections
	reconnaissance : ListDatasetEntries
	reconnaissance : ListDatasetLabels
	reconnaissance : ListFaces
	reconnaissance : ListMediaAnalysisJobs
	reconnaissance : ListProjectPolicies
	reconnaissance : ListStreamProcessors
	reconnaissance : ListUsers
	reconnaissance : PutProjectPolicy
	reconnaissance : RecognizeCelebrities
	reconnaissance : SearchFaces
	reconnaissance : SearchFacesByImage
	reconnaissance : SearchUsers
	reconnaissance : SearchUsersByImage
	reconnaissance : StartCelebrityRecognition
	reconnaissance : StartContentModeration
	reconnaissance : StartFaceDetection
	reconnaissance : StartFaceLivenessSession
	reconnaissance : StartFaceSearch
	reconnaissance : StartLabelDetection

Préfixe de service	Actions
	reconnaissance : StartMediaAnalysisJob
	reconnaissance : StartPersonTracking
	reconnaissance : StartProjectVersion
	reconnaissance : StartSegmentDetection
	reconnaissance : StartStreamProcessor
	reconnaissance : StartTextDetection
	reconnaissance : StopProjectVersion
	reconnaissance : StopStreamProcessor
	reconnaissance : UpdateDatasetEntries
	reconnaissance : UpdateStreamProcessor

Préfixe de service	Actions
resiliencehub	hub de résilience : AcceptResourceGroupingRecommendations hub de résilience : AddDraftAppVersionResourceMappings hub de résilience : BatchUpdateRecommendationStatus hub de résilience : CreateApp hub de résilience : CreateAppVersionAppComponent hub de résilience : CreateAppVersionResource hub de résilience : CreateRecommendationTemplate hub de résilience : CreateResiliencyPolicy hub de résilience : DeleteApp hub de résilience : DeleteAppAssessment hub de résilience : DeleteAppInputSource hub de résilience : DeleteAppVersionAppComponent hub de résilience : DeleteAppVersionResource hub de résilience : DeleteRecommendationTemplate hub de résilience : DeleteResiliencyPolicy hub de résilience : DescribeApp hub de résilience : DescribeAppAssessment hub de résilience : DescribeAppVersion hub de résilience : DescribeAppVersionAppComponent hub de résilience : DescribeAppVersionResource hub de résilience : DescribeAppVersionResourcesResolutionStatus

Préfixe de service	Actions
	hub de résilience : DescribeAppVersionTemplate hub de résilience : DescribeDraftAppVersionResourcesImportStatus hub de résilience : DescribeMetricsExport hub de résilience : DescribeResiliencyPolicy hub de résilience : DescribeResourceGroupingRecommendationTask hub de résilience : ImportResourcesToDraftAppVersion hub de résilience : ListAlarmRecommendations hub de résilience : ListAppAssessmentComplianceDrifts hub de résilience : ListAppAssessmentResourceDrifts hub de résilience : ListAppAssessments hub de résilience : ListAppComponentCompliances hub de résilience : ListAppComponentRecommendations hub de résilience : ListAppInputSources hub de résilience : ListAppVersionAppComponents hub de résilience : ListAppVersionResourceMappings hub de résilience : ListAppVersionResources hub de résilience : ListAppVersions hub de résilience : ListApps hub de résilience : ListMetrics hub de résilience : ListRecommendationTemplates

Préfixe de service	Actions
	hub de résilience : ListResiliencyPolicies hub de résilience : ListResourceGroupingRecommendations hub de résilience : ListSopRecommendations hub de résilience : ListSuggestedResiliencyPolicies hub de résilience : ListTestRecommendations hub de résilience : ListUnsupportedAppVersionResources hub de résilience : PublishAppVersion hub de résilience : PutDraftAppVersionTemplate hub de résilience : RejectResourceGroupingRecommendations hub de résilience : RemoveDraftAppVersionResourceMappings hub de résilience : ResolveAppVersionResources hub de résilience : StartAppAssessment hub de résilience : StartResourceGroupingRecommendationTask hub de résilience : UpdateApp hub de résilience : UpdateAppVersion hub de résilience : UpdateAppVersionAppComponent hub de résilience : UpdateAppVersionResource hub de résilience : UpdateResiliencyPolicy

Préfixe de service	Actions
resource-explorer-2	explorateur de ressources-2 : AssociateDefaultView
	explorateur de ressources-2 : BatchGetView
	explorateur de ressources-2 : CreateIndex
	explorateur de ressources-2 : CreateResourceExplorerSetup
	explorateur de ressources-2 : CreateView
	explorateur de ressources-2 : DeleteIndex
	explorateur de ressources-2 : DeleteResourceExplorerSetup
	explorateur de ressources-2 : DeleteView
	explorateur de ressources-2 : DisassociateDefaultView
	explorateur de ressources-2 : GetAccountLevelServiceConfiguration
	explorateur de ressources-2 : GetDefaultView
	explorateur de ressources-2 : GetIndex
	explorateur de ressources-2 : GetManagedView
	explorateur de ressources-2 : GetResourceExplorerSetup
	explorateur de ressources-2 : GetServiceIndex
	explorateur de ressources-2 : GetServiceView
	explorateur de ressources-2 : ListIndexes
	explorateur de ressources-2 : ListIndexesForMembers
	explorateur de ressources-2 : ListManagedViews
	explorateur de ressources-2 : ListServiceIndexes
	explorateur de ressources-2 : ListServiceViews

Préfixe de service	Actions
	explorateur de ressources-2 : ListStreamingAccessForServices
	explorateur de ressources-2 : ListSupportedResourceTypes
	explorateur de ressources-2 : ListViews
	resource-explorer-2:Search
	explorateur de ressources-2 : UpdateIndexType
	explorateur de ressources-2 : UpdateView

Préfixe de service	Actions
resource-groups	groupes de ressources : CancelTagSyncTask groupes de ressources : GetAccountSettings groupes de ressources : GetGroup groupes de ressources : GetGroupConfiguration groupes de ressources : GetGroupQuery groupes de ressources : GetTagSyncTask groupes de ressources : GroupResources groupes de ressources : ListGroupResources groupes de ressources : ListGroupingStatuses groupes de ressources : ListGroups groupes de ressources : ListTagSyncTasks groupes de ressources : PutGroupConfiguration groupes de ressources : SearchResources groupes de ressources : StartTagSyncTask groupes de ressources : UngroupResources groupes de ressources : UpdateAccountSettings groupes de ressources : UpdateGroup groupes de ressources : UpdateGroupQuery

Préfixe de service	Actions
robomaker	fabricant de robots : BatchDeleteWorlds
	fabricant de robots : BatchDescribeSimulationJob
	fabricant de robots : CancelDeploymentJob
	fabricant de robots : CancelSimulationJob
	fabricant de robots : CancelSimulationJobBatch
	fabricant de robots : CancelWorldExportJob
	fabricant de robots : CancelWorldGenerationJob
	fabricant de robots : CreateDeploymentJob
	fabricant de robots : CreateFleet
	fabricant de robots : CreateRobot
	fabricant de robots : CreateRobotApplication
	fabricant de robots : CreateRobotApplicationVersion
	fabricant de robots : CreateSimulationApplication
	fabricant de robots : CreateSimulationApplicationVersion
	fabricant de robots : CreateSimulationJob
	fabricant de robots : CreateWorldExportJob
	fabricant de robots : CreateWorldGenerationJob
	fabricant de robots : CreateWorldTemplate
	fabricant de robots : DeleteFleet
	fabricant de robots : DeleteRobot
	fabricant de robots : DeleteRobotApplication

Préfixe de service	Actions
	fabricant de robots : DeleteSimulationApplication
	fabricant de robots : DeleteWorldTemplate
	fabricant de robots : DeregisterRobot
	fabricant de robots : DescribeDeploymentJob
	fabricant de robots : DescribeFleet
	fabricant de robots : DescribeRobot
	fabricant de robots : DescribeRobotApplication
	fabricant de robots : DescribeSimulationApplication
	fabricant de robots : DescribeSimulationJob
	fabricant de robots : DescribeSimulationJobBatch
	fabricant de robots : DescribeWorld
	fabricant de robots : DescribeWorldExportJob
	fabricant de robots : DescribeWorldGenerationJob
	fabricant de robots : DescribeWorldTemplate
	fabricant de robots : GetWorldTemplateBody
	fabricant de robots : ListDeploymentJobs
	fabricant de robots : ListFleets
	fabricant de robots : ListRobotApplications
	fabricant de robots : ListRobots
	fabricant de robots : ListSimulationApplications
	fabricant de robots : ListSimulationJobBatches

Préfixe de service	Actions
	fabricant de robots : ListSimulationJobs
	fabricant de robots : ListWorldExportJobs
	fabricant de robots : ListWorldGenerationJobs
	fabricant de robots : ListWorldTemplates
	fabricant de robots : ListWorlds
	fabricant de robots : RegisterRobot
	fabricant de robots : RestartSimulationJob
	fabricant de robots : StartSimulationJobBatch
	fabricant de robots : SyncDeploymentJob
	fabricant de robots : UpdateRobotApplication
	fabricant de robots : UpdateSimulationApplication
	fabricant de robots : UpdateWorldTemplate

Préfixe de service	Actions
rolesanywhere	rôles n'importe où : CreateProfile rôles n'importe où : CreateTrustAnchor rôles n'importe où : DeleteAttributeMapping rôles n'importe où : DeleteCrl rôles n'importe où : DeleteProfile rôles n'importe où : DeleteTrustAnchor rôles n'importe où : DisableCrl rôles n'importe où : DisableProfile rôles n'importe où : DisableTrustAnchor rôles n'importe où : EnableCrl rôles n'importe où : EnableProfile rôles n'importe où : EnableTrustAnchor rôles n'importe où : GetCrl rôles n'importe où : GetProfile rôles n'importe où : GetSubject rôles n'importe où : GetTrustAnchor rôles n'importe où : ImportCrl rôles n'importe où : ListCrls rôles n'importe où : ListProfiles rôles n'importe où : ListSubjects rôles n'importe où : ListTrustAnchors

Préfixe de service	Actions
	rôles n'importe où : PutAttributeMapping rôles n'importe où : PutNotificationSettings rôles n'importe où : ResetNotificationSettings rôles n'importe où : UpdateCrl rôles n'importe où : UpdateProfile rôles n'importe où : UpdateTrustAnchor

Préfixe de service	Actions
route53	route 53 : ActivateKeySigningKey
	route 53 : AssociateVPCWithHostedZone
	route 53 : ChangeCidrCollection
	route 53 : ChangeResourceRecordSets
	route 53 : CreateCidrCollection
	route 53 : CreateHealthCheck
	route 53 : CreateHostedZone
	route 53 : CreateKeySigningKey
	route 53 : CreateQueryLoggingConfig
	route 53 : CreateReusableDelegationSet
	route 53 : CreateTrafficPolicy
	route 53 : CreateTrafficPolicyInstance
	route 53 : CreateTrafficPolicyVersion
	route 53 : CreateVPCAssociationAuthorization
	route 53 : DeactivateKeySigningKey
	route 53 : DeleteCidrCollection
	route 53 : DeleteHealthCheck
	route 53 : DeleteHostedZone
	route 53 : DeleteKeySigningKey
	route 53 : DeleteQueryLoggingConfig
	route 53 : DeleteReusableDelegationSet

Préfixe de service	Actions
	route 53 : DeleteTrafficPolicy
	route 53 : DeleteTrafficPolicyInstance
	route 53 : DeleteVPCAssociationAuthorization
	route 53 : DisableHostedZoneDNSSEC
	route 53 : DisassociateVPCFromHostedZone
	route 53 : EnableHostedZoneDNSSEC
	route 53 : GetAccountLimit
	route 53 : GetChange
	route 53 : GetCheckerIpRanges
	route53:GetDNSSEC
	route 53 : GetGeoLocation
	route 53 : GetHealthCheck
	route 53 : GetHealthCheckCount
	route 53 : GetHealthCheckLastFailureReason
	route 53 : GetHealthCheckStatus
	route 53 : GetHostedZone
	route 53 : GetHostedZoneCount
	route 53 : GetHostedZoneLimit
	route 53 : GetQueryLoggingConfig
	route 53 : GetReusableDelegationSet
	route 53 : GetReusableDelegationSetLimit

Préfixe de service	Actions
	route 53 : GetTrafficPolicy
	route 53 : GetTrafficPolicyInstance
	route 53 : GetTrafficPolicyInstanceCount
	route 53 : ListCidrBlocks
	route 53 : ListCidrCollections
	route 53 : ListCidrLocations
	route 53 : ListGeoLocations
	route 53 : ListHealthChecks
	route 53 : ListHostedZones
	route 53 : ListHostedZonesByName
	route 53 : ListHostedZonesByVPC
	route 53 : ListQueryLoggingConfigs
	route 53 : ListResourceRecordSets
	route 53 : ListReusableDelegationSets
	route 53 : ListTrafficPolicies
	route 53 : ListTrafficPolicyInstances
	route 53 : ListTrafficPolicyInstancesByHostedZone
	route 53 : ListTrafficPolicyInstancesByPolicy
	route 53 : ListTrafficPolicyVersions
	route 53 : ListVPCAssociationAuthorizations
	route53:TestDNSAnswer

Préfixe de service	Actions
	route 53 : UpdateHealthCheck route 53 : UpdateHostedZoneComment route 53 : UpdateTrafficPolicyComment route 53 : UpdateTrafficPolicyInstance

Préfixe de service	Actions
route53-recovery-control-config	route53-recovery-control-config : CreateCluster route53-recovery-control-config : CreateControlPanel route53-recovery-control-config : CreateRoutingControl route53-recovery-control-config : CreateSafetyRule route53-recovery-control-config : DeleteCluster route53-recovery-control-config : DeleteControlPanel route53-recovery-control-config : DeleteRoutingControl route53-recovery-control-config : DeleteSafetyRule route53-recovery-control-config : DescribeCluster route53-recovery-control-config : DescribeControlPanel route53-recovery-control-config : DescribeRoutingControl route53-recovery-control-config : DescribeSafetyRule route53-recovery-control-config : GetResourcePolicy route53-recovery-control-config : ListAssociatedRoute53HealthChecks route53-recovery-control-config : ListClusters route53-recovery-control-config : ListControlPanels route53-recovery-control-config : ListRoutingControls route53-recovery-control-config : ListSafetyRules route53-recovery-control-config : UpdateCluster route53-recovery-control-config : UpdateControlPanel

Préfixe de service	Actions
	route53-recovery-control-config : UpdateRoutingControl route53-recovery-control-config : UpdateSafetyRule

Préfixe de service	Actions
route53-recovery-readiness	route 53 - préparation au rétablissement : CreateCell
	route 53 - préparation au rétablissement : CreateCrossAccount Authorization
	route 53 - préparation au rétablissement : CreateReadinessCheck
	route 53 - préparation au rétablissement : CreateRecoveryGroup
	route 53 - préparation au rétablissement : CreateResourceSet
	route 53 - préparation au rétablissement : DeleteCell
	route 53 - préparation au rétablissement : DeleteCrossAccount Authorization
	route 53 - préparation au rétablissement : DeleteReadinessCheck
	route 53 - préparation au rétablissement : DeleteRecoveryGroup
	route 53 - préparation au rétablissement : DeleteResourceSet
	route 53 - préparation au rétablissement : GetArchitectureRecommendations
	route 53 - préparation au rétablissement : GetCell
	route 53 - préparation au rétablissement : GetCellReadinessSummary
	route 53 - préparation au rétablissement : GetReadinessCheck
	route 53 - préparation au rétablissement : GetReadinessCheckResourceStatus
	route 53 - préparation au rétablissement : GetReadinessCheckStatus
	route 53 - préparation au rétablissement : GetRecoveryGroup

Préfixe de service	Actions
	route 53 - préparation au rétablissement : GetRecoveryGroupReadinessSummary
	route 53 - préparation au rétablissement : GetResourceSet
	route 53 - préparation au rétablissement : ListCells
	route 53 - préparation au rétablissement : ListCrossAccountAuthorizations
	route 53 - préparation au rétablissement : ListReadinessChecks
	route 53 - préparation au rétablissement : ListRecoveryGroups
	route 53 - préparation au rétablissement : ListResourceSets
	route 53 - préparation au rétablissement : ListRules
	route 53 - préparation au rétablissement : UpdateCell
	route 53 - préparation au rétablissement : UpdateReadinessCheck
	route 53 - préparation au rétablissement : UpdateRecoveryGroup
	route 53 - préparation au rétablissement : UpdateResourceSet

Préfixe de service	Actions
route53resolver	résolveur Route53 : AssociateFirewallRuleGroup résolveur Route53 : AssociateResolverEndpointIpAddress résolveur Route53 : AssociateResolverQueryLogConfig résolveur Route53 : AssociateResolverRule résolveur Route53 : CreateFirewallDomainList résolveur Route53 : CreateFirewallRule résolveur Route53 : CreateFirewallRuleGroup résolveur Route53 : CreateResolverEndpoint résolveur Route53 : CreateResolverQueryLogConfig résolveur Route53 : CreateResolverRule résolveur Route53 : DeleteFirewallDomainList résolveur Route53 : DeleteFirewallRule résolveur Route53 : DeleteFirewallRuleGroup résolveur Route53 : DeleteOutpostResolver résolveur Route53 : DeleteResolverEndpoint résolveur Route53 : DeleteResolverQueryLogConfig résolveur Route53 : DeleteResolverRule résolveur Route53 : DisassociateFirewallRuleGroup résolveur Route53 : DisassociateResolverEndpointIpAddress résolveur Route53 : DisassociateResolverQueryLogConfig résolveur Route53 : DisassociateResolverRule

Préfixe de service	Actions
	résolveur Route53 : GetFirewallConfig résolveur Route53 : GetFirewallDomainList résolveur Route53 : GetFirewallRuleGroup résolveur Route53 : GetFirewallRuleGroupAssociation résolveur Route53 : GetFirewallRuleGroupPolicy résolveur Route53 : GetOutpostResolver résolveur Route53 : GetResolverConfig résolveur Route53 : GetResolverDnssecConfig résolveur Route53 : GetResolverEndpoint résolveur Route53 : GetResolverQueryLogConfig résolveur Route53 : GetResolverQueryLogConfigAssociation résolveur Route53 : GetResolverQueryLogConfigPolicy résolveur Route53 : GetResolverRule résolveur Route53 : GetResolverRuleAssociation résolveur Route53 : GetResolverRulePolicy résolveur Route53 : ImportFirewallDomains résolveur Route53 : ListFirewallConfigs résolveur Route53 : ListFirewallDomainLists résolveur Route53 : ListFirewallDomains résolveur Route53 : ListFirewallRuleGroupAssociations résolveur Route53 : ListFirewallRuleGroups

Préfixe de service	Actions
	résolveur Route53 : ListFirewallRuleTypes
	résolveur Route53 : ListFirewallRules
	résolveur Route53 : ListOutpostResolvers
	résolveur Route53 : ListResolverConfigs
	résolveur Route53 : ListResolverDnssecConfigs
	résolveur Route53 : ListResolverEndpointIpAddresses
	résolveur Route53 : ListResolverEndpoints
	résolveur Route53 : ListResolverQueryLogConfigAssociations
	résolveur Route53 : ListResolverQueryLogConfigs
	résolveur Route53 : ListResolverRuleAssociations
	résolveur Route53 : ListResolverRules
	résolveur Route53 : PutFirewallRuleGroupPolicy
	résolveur Route53 : PutResolverQueryLogConfigPolicy
	résolveur Route53 : UpdateFirewallConfig
	résolveur Route53 : UpdateFirewallDomains
	résolveur Route53 : UpdateFirewallRule
	résolveur Route53 : UpdateFirewallRuleGroupAssociation
	résolveur Route53 : UpdateOutpostResolver
	résolveur Route53 : UpdateResolverConfig
	résolveur Route53 : UpdateResolverDnssecConfig
	résolveur Route53 : UpdateResolverEndpoint

Préfixe de service	Actions
	résolveur Route53 : UpdateResolverRule
rum	rhum : BatchCreateRumMetricDefinitions
	rhum : BatchDeleteRumMetricDefinitions
	rhum : BatchGetRumMetricDefinitions
	rhum : CreateAppMonitor
	rhum : DeleteAppMonitor
	rhum : DeleteResourcePolicy
	rhum : DeleteRumMetricsDestination
	rhum : GetAppMonitor
	rhum : GetAppMonitorData
	rhum : GetResourcePolicy
	rhum : ListAppMonitors
	rhum : ListRumMetricsDestinations
	rhum : PutResourcePolicy
	rhum : PutRumMetricsDestination
	rhum : UpdateAppMonitor
	rhum : UpdateRumMetricDefinition

Préfixe de service	Actions
s3	s3 : AssociateAccessGrantsIdentityCenter s3 : CreateAccessGrant s3 : CreateAccessGrantsInstance s3 : CreateAccessGrantsLocation s3 : CreateAccessPoint s3 : CreateAccessPointForObjectLambda s3 : CreateBucket s3 : CreateBucketMetadataTableConfiguration s3 : CreateJob s3 : CreateMultiRegionAccessPoint s3 : DeleteAccessGrant s3 : DeleteAccessGrantsInstance s3 : DeleteAccessGrantsInstanceResourcePolicy s3 : DeleteAccessGrantsLocation s3 : DeleteAccessPoint s3 : DeleteAccessPointForObjectLambda s3 : DeleteAccessPointPolicy s3 : DeleteAccessPointPolicyForObjectLambda s3 : DeleteBucket s3 : DeleteBucketMetadataTableConfiguration s3 : DeleteBucketPolicy

Préfixe de service	Actions
	s3 : DeleteBucketWebsite s3 : DeleteMultiRegionAccessPoint s3 : DeleteStorageLensConfiguration s3 : DescribeJob s3 : DescribeMultiRegionAccessPointOperation s3 : DissociateAccessGrantsIdentityCenter s3 : GetAccelerateConfiguration s3 : GetAccessGrant s3 : GetAccessGrantsInstance s3 : GetAccessGrantsInstanceForPrefix s3 : GetAccessGrantsInstanceResourcePolicy s3 : GetAccessGrantsLocation s3 : GetAccessPoint s3 : GetAccessPointConfigurationForObjectLambda s3 : GetAccessPointForObjectLambda s3 : GetAccessPointPolicy s3 : GetAccessPointPolicyForObjectLambda s3 : GetAccessPointPolicyStatus s3 : GetAccessPointPolicyStatusForObjectLambda s3 : GetAccountPublicAccessBlock s3 : GetAnalyticsConfiguration

Préfixe de service	Actions
	s3 : GetBucketAbac s3 : GetBucketAcl s3 : GetBucket CORS s3 : GetBucketLocation s3 : GetBucketLogging s3 : GetBucketNotification s3 : GetBucketObjectLockConfiguration s3 : GetBucketOwnershipControls s3 : GetBucketPolicy s3 : GetBucketPolicyStatus s3 : GetBucketPublicAccessBlock s3 : GetBucketRequestPayment s3 : GetBucketVersioning s3 : GetBucketWebsite s3 : GetDataAccess s3 : GetEncryptionConfiguration s3 : GetIntelligentTieringConfiguration s3 : GetInventoryConfiguration s3 : GetLifecycleConfiguration s3 : GetMetricsConfiguration s3 : GetMultiRegionAccessPoint

Préfixe de service	Actions
	s3 : GetMultiRegionAccessPointPolicy s3 : GetMultiRegionAccessPointPolicyStatus s3 : GetMultiRegionAccessPointRoutes s3 : GetReplicationConfiguration s3 : GetStorageLensConfiguration s3 : GetStorageLensDashboard s3 : ListAccessGrants s3 : ListAccessGrantsInstances s3 : ListAccessGrantsLocations s3 : ListAccessPoints s3 : ListAccessPointsForObjectLambda s3 : ListAllMyBuckets s3 : ListBucketMultipartUploads s3 : ListCallerAccessGrants s3 : ListJobs s3 : ListMultiRegionAccessPoints s3 : ListStorageLensConfigurations s3 : PutAccelerateConfiguration s3 : PutAccessGrantsInstanceResourcePolicy s3 : PutAccessPointConfigurationForObjectLambda s3 : PutAccessPointPolicy

Préfixe de service	Actions
	s3 : PutAccessPointPolicyForObjectLambda s3 : PutAccountPublicAccessBlock s3 : PutAnalyticsConfiguration s3 : PutBucketAbac s3 : PutBucketAcl s3 : PutBucket CORS s3 : PutBucketLogging s3 : PutBucketNotification s3 : PutBucketObjectLockConfiguration s3 : PutBucketOwnershipControls s3 : PutBucketPolicy s3 : PutBucketPublicAccessBlock s3 : PutBucketRequestPayment s3 : PutBucketVersioning s3 : PutBucketWebsite s3 : PutEncryptionConfiguration s3 : PutIntelligentTieringConfiguration s3 : PutInventoryConfiguration s3 : PutLifecycleConfiguration s3 : PutMetricsConfiguration s3 : PutMultiRegionAccessPointPolicy

Préfixe de service	Actions
	s3 : PutReplicationConfiguration s3 : PutStorageLensConfiguration s3 : SubmitMultiRegionAccessPointRoutes s3 : UpdateAccessGrantsLocation s3 : UpdateBucketMetadataAnnotationTableConfiguration s3 : UpdateBucketMetadataJournalTableConfiguration s3 : UpdateJobPriority s3 : UpdateJobStatus
s3-outposts	avant-postes S3 : CreateEndpoint avant-postes S3 : DeleteEndpoint avant-postes S3 : ListEndpoints avant-postes S3 : ListOutpostsWithS3 avant-postes S3 : ListSharedEndpoints

Préfixe de service	Actions
sagemaker-geospatial	sagemaker-geospatial : DeleteEarthObservationJob sagemaker-geospatial : DeleteVectorEnrichmentJob sagemaker-geospatial : ExportEarthObservationJob sagemaker-geospatial : ExportVectorEnrichmentJob sagemaker-geospatial : GetEarthObservationJob sagemaker-geospatial : GetRasterDataCollection sagemaker-geospatial : GetTile sagemaker-geospatial : GetVectorEnrichmentJob sagemaker-geospatial : ListEarthObservationJobs sagemaker-geospatial : ListRasterDataCollections sagemaker-geospatial : ListVectorEnrichmentJobs sagemaker-geospatial : SearchRasterDataCollection sagemaker-geospatial : StartEarthObservationJob sagemaker-geospatial : StartVectorEnrichmentJob sagemaker-geospatial : StopEarthObservationJob sagemaker-geospatial : StopVectorEnrichmentJob

Préfixe de service	Actions
savingsplans	plans d'épargne : CreateSavingsPlan
	plans d'épargne : DeleteQueuedSavingsPlan
	plans d'épargne : DescribeSavingsPlanRates
	plans d'épargne : DescribeSavingsPlans
	plans d'épargne : DescribeSavingsPlansOfferingRates
	plans d'épargne : DescribeSavingsPlansOfferings
	plans d'épargne : ReturnSavingsPlan

Préfixe de service	Actions
schemas	schémas : CreateDiscoverer schémas : CreateRegistry schémas : CreateSchema schémas : DeleteDiscoverer schémas : DeleteRegistry schémas : DeleteResourcePolicy schémas : DeleteSchema schémas : DeleteSchemaVersion schémas : DescribeCodeBinding schémas : DescribeDiscoverer schémas : DescribeRegistry schémas : DescribeSchema schémas : ExportSchema schémas : GetCodeBindingSource schémas : GetDiscoveredSchema schémas : GetResourcePolicy schémas : ListDiscoverers schémas : ListRegistries schémas : ListSchemaVersions schémas : ListSchemas schémas : PutCodeBinding

Préfixe de service	Actions
	schémas : PutResourcePolicy schémas : SearchSchemas schémas : StartDiscoverer schémas : StopDiscoverer schémas : UpdateDiscoverer schémas : UpdateRegistry schémas : UpdateSchema
sdb	sdb : CreateDomain sdb : DeleteDomain sdb : DomainMetadata sdb : ListDomains

Préfixe de service	Actions
secretsmanager	responsable des secrets : CancelRotateSecret responsable des secrets : CreateSecret responsable des secrets : DeleteResourcePolicy responsable des secrets : DeleteSecret responsable des secrets : DescribeSecret responsable des secrets : GetRandomPassword responsable des secrets : GetResourcePolicy responsable des secrets : GetSecretValue responsable des secrets : ListSecretVersionIds responsable des secrets : ListSecrets responsable des secrets : PutResourcePolicy responsable des secrets : PutSecretValue responsable des secrets : RemoveRegionsFromReplication responsable des secrets : ReplicateSecretToRegions responsable des secrets : RestoreSecret responsable des secrets : RotateSecret responsable des secrets : StopReplicationToReplica responsable des secrets : UpdateSecret responsable des secrets : ValidateResourcePolicy

Préfixe de service	Actions
securityhub	hub de sécurité : AcceptAdministratorInvitation
	hub de sécurité : AcceptInvitation
	hub de sécurité : BatchDeleteAutomationRules
	hub de sécurité : BatchDisableStandards
	hub de sécurité : BatchEnableStandards
	hub de sécurité : BatchGetAutomationRules
	hub de sécurité : BatchGetConfigurationPolicyAssociations
	hub de sécurité : BatchGetEnabledRegionsV2
	hub de sécurité : BatchGetSecurityControls
	hub de sécurité : BatchGetStandardsControlAssociations
	hub de sécurité : BatchImportFindings
	hub de sécurité : BatchUpdateAutomationRules
	hub de sécurité : BatchUpdateFindings
	hub de sécurité : BatchUpdateStandardsControlAssociations
	hub de sécurité : V2 ConnectorRegistrations
	hub de sécurité : CreateActionTarget
	hub de sécurité : V2 CreateAggregator
	hub de sécurité : CreateAutomationRule
	hub de sécurité : CreateAutomationRuleV2
	hub de sécurité : CreateConfigurationPolicy
	hub de sécurité : CreateConnector

Préfixe de service	Actions
	hub de sécurité : V2 CreateConnector
	hub de sécurité : CreateFindingAggregator
	hub de sécurité : CreateInsight
	hub de sécurité : CreateMembers
	hub de sécurité : V2 CreateTicket
	hub de sécurité : DeclineInvitations
	hub de sécurité : DeleteActionTarget
	hub de sécurité : V2 DeleteAggregator
	hub de sécurité : DeleteAutomationRuleV2
	hub de sécurité : DeleteConfigurationPolicy
	hub de sécurité : DeleteConnector
	hub de sécurité : V2 DeleteConnector
	hub de sécurité : DeleteFindingAggregator
	hub de sécurité : DeleteInsight
	hub de sécurité : DeleteInvitations
	hub de sécurité : DeleteMembers
	hub de sécurité : DescribeActionTargets
	hub de sécurité : DescribeHub
	hub de sécurité : DescribeOrganizationConfiguration
	hub de sécurité : DescribeProducts
	hub de sécurité : DescribeSecurityHubV2

Préfixe de service	Actions
	hub de sécurité : DescribeStandards
	hub de sécurité : DisableImportFindingsForProduct
	hub de sécurité : DisableOrganizationAdminAccount
	hub de sécurité : DisableSecurityHub
	hub de sécurité : DisableSecurityHubFeatureV2
	hub de sécurité : DisableSecurityHubV2
	hub de sécurité : DisassociateFromAdministratorAccount
	hub de sécurité : DisassociateFromMasterAccount
	hub de sécurité : DisassociateMembers
	hub de sécurité : EnableImportFindingsForProduct
	hub de sécurité : EnableOrganizationAdminAccount
	hub de sécurité : EnableSecurityHub
	hub de sécurité : EnableSecurityHubFeatureV2
	hub de sécurité : GetAdministratorAccount
	hub de sécurité : V2 GetAggregator
	hub de sécurité : GetAutomationRuleV2
	hub de sécurité : GetConfigurationPolicy
	hub de sécurité : GetConfigurationPolicyAssociation
	hub de sécurité : GetConnector
	hub de sécurité : V2 GetConnector
	hub de sécurité : GetCoverageStatisticsV2

Préfixe de service	Actions
	hub de sécurité : GetEnabledStandards
	hub de sécurité : GetFindingAggregator
	hub de sécurité : GetFindingHistory
	hub de sécurité : GetFindings
	hub de sécurité : GetInsightResults
	hub de sécurité : GetInsights
	hub de sécurité : GetInvitationsCount
	hub de sécurité : GetMasterAccount
	hub de sécurité : GetMembers
	hub de sécurité : GetSecurityControlDefinition
	hub de sécurité : InviteMembers
	hub de sécurité : V2 ListAggregators
	hub de sécurité : ListAutomationRules
	hub de sécurité : ListAutomationRulesV2
	hub de sécurité : ListConfigurationPolicies
	hub de sécurité : ListConfigurationPolicyAssociations
	hub de sécurité : ListConnectors
	hub de sécurité : V2 ListConnectors
	hub de sécurité : ListEnabledProductsForImport
	hub de sécurité : ListFindingAggregators
	hub de sécurité : ListInvitations

Préfixe de service	Actions
	hub de sécurité : ListMembers
	hub de sécurité : ListOrganizationAdminAccounts
	hub de sécurité : ListSecurityControlDefinitions
	hub de sécurité : ListStandardsControlAssociations
	hub de sécurité : StartConfigurationPolicyAssociation
	hub de sécurité : StartConfigurationPolicyDisassociation
	hub de sécurité : UpdateActionTarget
	hub de sécurité : V2 UpdateAggregator
	hub de sécurité : UpdateAutomationRuleV2
	hub de sécurité : UpdateConfigurationPolicy
	hub de sécurité : UpdateConnector
	hub de sécurité : V2 UpdateConnector
	hub de sécurité : UpdateFindingAggregator
	hub de sécurité : UpdateFindings
	hub de sécurité : UpdateInsight
	hub de sécurité : UpdateOrganizationConfiguration
	hub de sécurité : UpdateSecurityControl
	hub de sécurité : UpdateSecurityHubConfiguration

Préfixe de service	Actions
securitylake	lac de sécurité : CreateAwsLogSource lac de sécurité : CreateCustomLogSource lac de sécurité : CreateDataLakeExceptionSubscription lac de sécurité : CreateDataLakeOrganizationConfiguration lac de sécurité : CreateSubscriber lac de sécurité : CreateSubscriberNotification lac de sécurité : DeleteAwsLogSource lac de sécurité : DeleteCustomLogSource lac de sécurité : DeleteDataLakeExceptionSubscription lac de sécurité : DeleteDataLakeOrganizationConfiguration lac de sécurité : DeleteSubscriber lac de sécurité : DeleteSubscriberNotification lac de sécurité : DeregisterDataLakeDelegatedAdministrator lac de sécurité : GetDataLakeExceptionSubscription lac de sécurité : GetDataLakeOrganizationConfiguration lac de sécurité : GetDataLakeSources lac de sécurité : GetSubscriber lac de sécurité : ListDataLakes lac de sécurité : ListLogSources lac de sécurité : ListSubscribers lac de sécurité : RegisterDataLakeDelegatedAdministrator

Préfixe de service	Actions
	lac de sécurité : UpdateDataLakeExceptionSubscription lac de sécurité : UpdateSubscriber lac de sécurité : UpdateSubscriberNotification
serverlessrepo	dépôt sans serveur : CreateApplication dépôt sans serveur : CreateApplicationVersion dépôt sans serveur : CreateCloudFormationChangeSet dépôt sans serveur : CreateCloudFormationTemplate dépôt sans serveur : DeleteApplication dépôt sans serveur : GetApplication dépôt sans serveur : GetApplicationPolicy dépôt sans serveur : GetCloudFormationTemplate dépôt sans serveur : ListApplicationDependencies dépôt sans serveur : ListApplicationVersions dépôt sans serveur : ListApplications dépôt sans serveur : PutApplicationPolicy dépôt sans serveur : UnshareApplication dépôt sans serveur : UpdateApplication

Préfixe de service	Actions
servicecatalog	catalogue de services : AcceptPortfolioShare catalogue de services : AssociateBudgetWithResource catalogue de services : AssociatePrincipalWithPortfolio catalogue de services : AssociateProductWithPortfolio catalogue de services : AssociateServiceActionWithProvisioningArtifact catalogue de services : BatchAssociateServiceActionWithProvisioningArtifact catalogue de services : BatchDisassociateServiceActionFromProvisioningArtifact catalogue de services : CopyProduct catalogue de services : CreateAttributeGroup catalogue de services : CreateConstraint catalogue de services : CreatePortfolio catalogue de services : CreatePortfolioShare catalogue de services : CreateProduct catalogue de services : CreateProvisionedProductPlan catalogue de services : CreateProvisioningArtifact catalogue de services : CreateServiceAction catalogue de services : DeleteAttributeGroup catalogue de services : DeleteConstraint catalogue de services : DeletePortfolio

Préfixe de service	Actions
	catalogue de services : DeletePortfolioShare catalogue de services : DeleteProduct catalogue de services : DeleteProvisionedProductPlan catalogue de services : DeleteProvisioningArtifact catalogue de services : DeleteServiceAction catalogue de services : DescribeConstraint catalogue de services : DescribeCopyProductStatus catalogue de services : DescribePortfolio catalogue de services : DescribePortfolioShareStatus catalogue de services : DescribePortfolioShares catalogue de services : DescribeProduct catalogue de services : DescribeProductAsAdmin catalogue de services : DescribeProductView catalogue de services : DescribeProvisionedProduct catalogue de services : DescribeProvisionedProductPlan catalogue de services : DescribeProvisioningArtifact catalogue de services : DescribeProvisioningParameters catalogue de services : DescribeRecord catalogue de services : DescribeServiceAction catalogue de services : DescribeServiceActionExecutionParameters catalogue de services : DisableAWSOrganizationsAccess

Préfixe de service	Actions
	catalogue de services : DisassociateBudgetFromResource catalogue de services : DisassociatePrincipalFromPortfolio catalogue de services : DisassociateProductFromPortfolio catalogue de services : DisassociateServiceActionFromProvisioningArtifact catalogue de services : EnableAWSOrganizationsAccess catalogue de services : ExecuteProvisionedProductPlan catalogue de services : ExecuteProvisionedProductServiceAction catalogue de services : GetAWSOrganizationsAccessStatus catalogue de services : GetProvisionedProductOutputs catalogue de services : ImportAsProvisionedProduct catalogue de services : ListAcceptedPortfolioShares catalogue de services : ListAttributeGroups catalogue de services : ListBudgetsForResource catalogue de services : ListConstraintsForPortfolio catalogue de services : ListLaunchPaths catalogue de services : ListOrganizationPortfolioAccess catalogue de services : ListPortfolioAccess catalogue de services : ListPortfolios catalogue de services : ListPortfoliosForProduct catalogue de services : ListPrincipalsForPortfolio

Préfixe de service	Actions
	catalogue de services : ListProvisionedProductPlans catalogue de services : ListProvisioningArtifacts catalogue de services : ListProvisioningArtifactsForServiceAction catalogue de services : ListRecordHistory catalogue de services : ListServiceActions catalogue de services : ListServiceActionsForProvisioningArtifact catalogue de services : ListStackInstancesForProvisionedProduct catalogue de services : NotifyProvisionProductEngineWorkflow Result catalogue de services : NotifyTerminateProvisionedProductEng ineWorkflowResult catalogue de services : NotifyUpdateProvisionedProductEngine WorkflowResult catalogue de services : ProvisionProduct catalogue de services : RejectPortfolioShare catalogue de services : ScanProvisionedProducts catalogue de services : SearchProducts catalogue de services : SearchProductsAsAdmin catalogue de services : SearchProvisionedProducts catalogue de services : TerminateProvisionedProduct catalogue de services : UpdateConstraint catalogue de services : UpdatePortfolio

Préfixe de service	Actions
	catalogue de services : UpdatePortfolioShare catalogue de services : UpdateProduct catalogue de services : UpdateProvisionedProduct catalogue de services : UpdateProvisionedProductProperties catalogue de services : UpdateProvisioningArtifact catalogue de services : UpdateServiceAction

Préfixe de service	Actions
servicediscovery	découverte des services : CreateHttpNamespace
	découverte des services : CreatePrivateDnsNamespace
	découverte des services : CreatePublicDnsNamespace
	découverte des services : CreateService
	découverte des services : DeleteNamespace
	découverte des services : DeleteService
	découverte des services : DeleteServiceAttributes
	découverte des services : DeregisterInstance
	découverte des services : GetInstance
	découverte des services : GetInstancesHealthStatus
	découverte des services : GetNamespace
	découverte des services : GetOperation
	découverte des services : GetService
	découverte des services : ListInstances
	découverte des services : ListNamespaces
	découverte des services : ListOperations
	découverte des services : ListServices
	découverte des services : RegisterInstance
	découverte des services : UpdateHttpNamespace
	découverte des services : UpdateInstanceCustomHealthStatus
	découverte des services : UpdatePrivateDnsNamespace

Préfixe de service	Actions
	découverte des services : UpdatePublicDnsNamespace découverte des services : UpdateService découverte des services : UpdateServiceAttributes

Préfixe de service	Actions
servicequotas	devis de service : AssociateServiceQuotaTemplate devis de service : CreateSupportCase devis de service : DeleteServiceQuotaIncreaseRequestFromTemplate devis de service : DisassociateServiceQuotaTemplate devis de service : GetAWSDefaultServiceQuota devis de service : GetAssociationForServiceQuotaTemplate devis de service : GetAutoManagementConfiguration devis de service : GetQuotaUtilizationReport devis de service : GetRequestedServiceQuotaChange devis de service : GetServiceQuota devis de service : GetServiceQuotaIncreaseRequestFromTemplate devis de service : ListAWSDefaultServiceQuotas devis de service : ListRequestedServiceQuotaChangeHistory devis de service : ListRequestedServiceQuotaChangeHistoryByQuota devis de service : ListServiceQuotaIncreaseRequestsInTemplate devis de service : ListServiceQuotas devis de service : ListServices devis de service : PutServiceQuotaIncreaseRequestIntoTemplate devis de service : RequestServiceQuotaIncrease devis de service : StartAutoManagement

Préfixe de service	Actions
	devis de service : StartQuotaUtilizationReport devis de service : StopAutoManagement devis de service : UpdateAutoManagement

Préfixe de service	Actions
ses	Utilise : BatchGetMetricData
	Utilise : CloneReceiptRuleSet
	Utilise : CreateAddonInstance
	Utilise : CreateAddonSubscription
	Utilise : CreateAddressList
	Utilise : CreateAddressListImportJob
	Utilise : CreateArchive
	Utilise : CreateConfigurationSet
	Utilise : CreateConfigurationSetEventDestination
	Utilise : CreateConfigurationSetTrackingOptions
	Utilise : CreateContact
	Utilise : CreateContactList
	Utilise : CreateCustomVerificationEmailTemplate
	Utilise : CreateDedicatedIpPool
	Utilise : CreateDeliverabilityTestReport
	Utilise : CreateEmailIdentity
	Utilise : CreateEmailIdentityPolicy
	Utilise : CreateEmailTemplate
	Utilise : CreateImportJob
	Utilise : CreateIngressPoint
	Utilise : CreateMultiRegionEndpoint

Préfixe de service	Actions
	Utilise : CreateReceiptFilter
	Utilise : CreateReceiptRule
	Utilise : CreateReceiptRuleSet
	Utilise : CreateRelay
	Utilise : CreateRuleSet
	Utilise : CreateTemplate
	Utilise : CreateTenant
	Utilise : CreateTenantResourceAssociation
	Utilise : CreateTrafficPolicy
	Utilise : DeleteAddonInstance
	Utilise : DeleteAddonSubscription
	Utilise : DeleteAddressList
	Utilise : DeleteArchive
	Utilise : DeleteConfigurationSet
	Utilise : DeleteConfigurationSetEventDestination
	Utilise : DeleteConfigurationSetTrackingOptions
	Utilise : DeleteContact
	Utilise : DeleteContactList
	Utilise : DeleteCustomVerificationEmailTemplate
	Utilise : DeleteDedicatedIpPool
	Utilise : DeleteEmailIdentity

Préfixe de service	Actions
	Utilise : DeleteEmailIdentityPolicy
	Utilise : DeleteEmailTemplate
	Utilise : DeleteIdentity
	Utilise : DeleteIdentityPolicy
	Utilise : DeleteIngressPoint
	Utilise : DeleteMultiRegionEndpoint
	Utilise : DeleteReceiptFilter
	Utilise : DeleteReceiptRule
	Utilise : DeleteReceiptRuleSet
	Utilise : DeleteRelay
	Utilise : DeleteRuleSet
	Utilise : DeleteSuppressedDestination
	Utilise : DeleteTemplate
	Utilise : DeleteTenant
	Utilise : DeleteTenantResourceAssociation
	Utilise : DeleteTrafficPolicy
	Utilise : DeleteVerifiedEmailAddress
	Utilise : DeregisterMemberFromAddressList
	Utilise : DescribeActiveReceiptRuleSet
	Utilise : DescribeConfigurationSet
	Utilise : DescribeReceiptRule

Préfixe de service	Actions
	Utilise : DescribeReceiptRuleSet
	Utilise : GetAccount
	Utilise : GetAccountSendingEnabled
	Utilise : GetAddonInstance
	Utilise : GetAddonSubscription
	Utilise : GetAddressList
	Utilise : GetArchive
	Utilise : GetArchiveExport
	Utilise : GetArchiveMessage
	Utilise : GetArchiveMessageContent
	Utilise : GetArchiveSearch
	Utilise : GetArchiveSearchResults
	Utilise : GetBlacklistReports
	Utilise : GetConfigurationSet
	Utilise : GetConfigurationSetEventDestinations
	Utilise : GetContact
	Utilise : GetContactList
	Utilise : GetCustomVerificationEmailTemplate
	Utilise : GetDedicatedIp
	Utilise : GetDedicatedIpPool
	Utilise : GetDedicatedIps

Préfixe de service	Actions
	Utilise : GetDeliverabilityDashboardOptions
	Utilise : GetDeliverabilityTestReport
	Utilise : GetDomainDeliverabilityCampaign
	Utilise : GetDomainStatisticsReport
	Utilise : GetEmailAddressInsights
	Utilise : GetEmailIdentity
	Utilise : GetEmailIdentityPolicies
	Utilise : GetEmailTemplate
	Utilise : GetIdentityDkimAttributes
	Utilise : GetIdentityMailFromDomainAttributes
	Utilise : GetIdentityNotificationAttributes
	Utilise : GetIdentityPolicies
	Utilise : GetIdentityVerificationAttributes
	Utilise : GetImportJob
	Utilise : GetIngressPoint
	Utilise : GetMemberOfAddressList
	Utilise : GetMessageInsights
	Utilise : GetMultiRegionEndpoint
	Utilise : GetRelay
	Utilise : GetRuleSet
	Utilise : GetSendQuota

Préfixe de service	Actions
	Utilise : GetSendStatistics
	Utilise : GetSuppressedDestination
	Utilise : GetTemplate
	Utilise : GetTenant
	Utilise : GetTrafficPolicy
	Utilise : ListAddonInstances
	Utilise : ListAddonSubscriptions
	Utilise : ListAddressListImportJobs
	Utilise : ListAddressLists
	Utilise : ListArchiveExports
	Utilise : ListArchiveSearches
	Utilise : ListArchives
	Utilise : ListConfigurationSets
	Utilise : ListContactLists
	Utilise : ListContacts
	Utilise : ListCustomVerificationEmailTemplates
	Utilise : ListDedicatedIpPools
	Utilise : ListDeliverabilityTestReports
	Utilise : ListDomainDeliverabilityCampaigns
	Utilise : ListEmailIdentities
	Utilise : ListEmailTemplates

Préfixe de service	Actions
	Utilise : ListExportJobs
	Utilise : ListIdentities
	Utilise : ListIdentityPolicies
	Utilise : ListImportJobs
	Utilise : ListIngressPoints
	Utilise : ListMembersOfAddressList
	Utilise : ListMultiRegionEndpoints
	Utilise : ListReceiptFilters
	Utilise : ListReceiptRuleSets
	Utilise : ListRecommendations
	Utilise : ListRelays
	Utilise : ListReputationEntities
	Utilise : ListResourceTenants
	Utilise : ListRuleSets
	Utilise : ListSuppressedDestinations
	Utilise : ListTemplates
	Utilise : ListTenantResources
	Utilise : ListTenants
	Utilise : ListTrafficPolicies
	Utilise : ListVerifiedEmailAddresses
	Utilise : PutAccountDedicatedIpWarmupAttributes

Préfixe de service	Actions
	Utilise : PutAccountDetails
	Utilise : PutAccountSendingAttributes
	Utilise : PutAccountSuppressionAttributes
	Utilise : PutAccountVdmAttributes
	Utilise : PutConfigurationSetArchivingOptions
	Utilise : PutConfigurationSetDeliveryOptions
	Utilise : PutConfigurationSetReputationOptions
	Utilise : PutConfigurationSetSendingOptions
	Utilise : PutConfigurationSetSuppressionOptions
	Utilise : PutConfigurationSetTrackingOptions
	Utilise : PutConfigurationSetVdmOptions
	Utilise : PutDedicatedIpInPool
	Utilise : PutDedicatedIpPoolScalingAttributes
	Utilise : PutDedicatedIpWarmupAttributes
	Utilise : PutDeliverabilityDashboardOption
	Utilise : PutEmailIdentityConfigurationSetAttributes
	Utilise : PutEmailIdentityDkimAttributes
	Utilise : PutEmailIdentityDkimSigningAttributes
	Utilise : PutEmailIdentityFeedbackAttributes
	Utilise : PutEmailIdentityMailFromAttributes
	Utilise : PutIdentityPolicy

Préfixe de service	Actions
	Utilise : PutSuppressedDestination
	Utilise : PutTenantSuppressionAttributes
	Utilise : RegisterMemberToAddressList
	Utilise : ReorderReceiptRuleSet
	Utilise : SendBounce
	Utilise : SendCustomVerificationEmail
	Utilise : SetActiveReceiptRuleSet
	Utilise : SetIdentityDkimEnabled
	Utilise : SetIdentityFeedbackForwardingEnabled
	Utilise : SetIdentityHeadersInNotificationsEnabled
	Utilise : SetIdentityMailFromDomain
	Utilise : SetIdentityNotificationTopic
	Utilise : SetReceiptRulePosition
	Utilise : StartArchiveExport
	Utilise : StartArchiveSearch
	Utilise : StopArchiveExport
	Utilise : StopArchiveSearch
	Utilise : TestRenderEmailTemplate
	Utilise : TestRenderTemplate
	Utilise : UpdateAccountSendingEnabled
	Utilise : UpdateArchive

Préfixe de service	Actions
	Utilise : UpdateConfigurationSetEventDestination
	Utilise : UpdateConfigurationSetReputationMetricsEnabled
	Utilise : UpdateConfigurationSetSendingEnabled
	Utilise : UpdateConfigurationSetTrackingOptions
	Utilise : UpdateContact
	Utilise : UpdateContactList
	Utilise : UpdateCustomVerificationEmailTemplate
	Utilise : UpdateEmailIdentityPolicy
	Utilise : UpdateEmailTemplate
	Utilise : UpdateIngressPoint
	Utilise : UpdateReceiptRule
	Utilise : UpdateRelay
	Utilise : UpdateRuleSet
	Utilise : UpdateTemplate
	Utilise : UpdateTrafficPolicy
	Utilise : VerifyDomainDkim
	Utilise : VerifyDomainIdentity
	Utilise : VerifyEmailAddress
	Utilise : VerifyEmailIdentity

Préfixe de service	Actions
shield	bouclier : AssociateDRTLogBucket bouclier : AssociateHealthCheck bouclier : AssociateProactiveEngagementDetails bouclier : CreateProtection bouclier : CreateProtectionGroup bouclier : CreateSubscription bouclier : DeleteProtection bouclier : DeleteProtectionGroup bouclier : DeleteSubscription bouclier : DescribeAttack bouclier : DescribeAttackStatistics shield:DescribeDRTAccess bouclier : DescribeEmergencyContactSettings bouclier : DescribeProtection bouclier : DescribeProtectionGroup bouclier : DescribeSubscription bouclier : DisableApplicationLayerAutomaticResponse bouclier : DisableProactiveEngagement bouclier : DisassociateDRTLogBucket shield:DisassociateDRTRole bouclier : DisassociateHealthCheck

Préfixe de service	Actions
	bouclier : EnableApplicationLayerAutomaticResponse bouclier : EnableProactiveEngagement bouclier : GetSubscriptionState bouclier : ListAttacks bouclier : ListProtectionGroups bouclier : ListProtections bouclier : ListResourcesInProtectionGroup bouclier : UpdateApplicationLayerAutomaticResponse bouclier : UpdateEmergencyContactSettings bouclier : UpdateProtectionGroup bouclier : UpdateSubscription

Préfixe de service	Actions
signer	signataire : AddProfilePermission signataire : CancelSigningProfile signataire : DescribeSigningJob signataire : GetSigningPlatform signataire : GetSigningProfile signataire : ListProfilePermissions signataire : ListSigningJobs signataire : ListSigningPlatforms signataire : ListSigningProfiles signataire : PutSigningProfile signataire : RemoveProfilePermission signataire : RevokeSignature signataire : RevokeSigningProfile signataire : SignPayload signataire : StartSigningJob

Préfixe de service	Actions
simspaceweaver	Simspaceweaver : CreateSnapshot Simspaceweaver : DeleteApp Simspaceweaver : DeleteSimulation Simspaceweaver : DescribeApp Simspaceweaver : DescribeSimulation Simspaceweaver : ListApps Simspaceweaver : ListSimulations Simspaceweaver : StartApp Simspaceweaver : StartClock Simspaceweaver : StartSimulation Simspaceweaver : StopApp Simspaceweaver : StopClock Simspaceweaver : StopSimulation

Préfixe de service	Actions
sms	SMS : CreateApp SMS : CreateReplicationJob SMS : DeleteApp SMS : DeleteAppLaunchConfiguration SMS : DeleteAppReplicationConfiguration SMS : DeleteAppValidationConfiguration SMS : DeleteReplicationJob SMS : DeleteServerCatalog SMS : DisassociateConnector SMS : GenerateChangeSet SMS : GenerateTemplate SMS : GetApp SMS : GetAppLaunchConfiguration SMS : GetAppReplicationConfiguration SMS : GetAppValidationConfiguration SMS : GetAppValidationOutput SMS : GetConnectors SMS : GetReplicationJobs SMS : GetReplicationRuns SMS : GetServers SMS : ImportAppCatalog

Préfixe de service	Actions
	SMS : ImportServerCatalog
	SMS : LaunchApp
	SMS : ListApps
	SMS : NotifyAppValidationOutput
	SMS : PutAppLaunchConfiguration
	SMS : PutAppReplicationConfiguration
	SMS : PutAppValidationConfiguration
	SMS : StartAppReplication
	SMS : StartOnDemandAppReplication
	SMS : StartOnDemandReplicationRun
	SMS : StopAppReplication
	SMS : TerminateApp
	SMS : UpdateApp
	SMS : UpdateReplicationJob

Préfixe de service	Actions
sms-voice	SMS et voix : AssociateProtectConfiguration SMS et voix : CreateConfigurationSet SMS et voix : CreateConfigurationSetEventDestination SMS et voix : CreateEventDestination SMS et voix : CreateOptOutList SMS et voix : CreatePool SMS et voix : CreateProtectConfiguration SMS et voix : CreateRegistration SMS et voix : CreateRegistrationAssociation SMS et voix : CreateRegistrationAttachment SMS et voix : CreateRegistrationVersion SMS et voix : CreateVerifiedDestinationNumber SMS et voix : DeleteAccountDefaultProtectConfiguration SMS et voix : DeleteConfigurationSet SMS et voix : DeleteConfigurationSetEventDestination SMS et voix : DeleteDefaultMessageType SMS et voix : DeleteDefaultSenderId SMS et voix : DeleteEventDestination SMS et voix : DeleteKeyword SMS et voix : DeleteMediaMessageSpendLimitOverride SMS et voix : DeleteOptOutList

Préfixe de service	Actions
	SMS et voix : DeleteOptedOutNumber SMS et voix : DeletePool SMS et voix : DeleteProtectConfiguration SMS et voix : DeleteProtectConfigurationRuleSetNumberOverride SMS et voix : DeleteRegistration SMS et voix : DeleteRegistrationAttachment SMS et voix : DeleteResourcePolicy SMS et voix : DeleteTextMessageSpendLimitOverride SMS et voix : DeleteVerifiedDestinationNumber SMS et voix : DeleteVoiceMessageSpendLimitOverride SMS et voix : DescribeAccountAttributes SMS et voix : DescribeAccountLimits SMS et voix : DescribeConfigurationSets SMS et voix : DescribeKeywords SMS et voix : DescribeOptOutLists SMS et voix : DescribeOptedOutNumbers SMS et voix : DescribePhoneNumbers SMS et voix : DescribePools SMS et voix : DescribeProtectConfigurations SMS et voix : DescribeRegistrationAttachments SMS et voix : DescribeRegistrationFieldDefinitions

Préfixe de service	Actions
	SMS et voix : DescribeRegistrationFieldValues
	SMS et voix : DescribeRegistrationSectionDefinitions
	SMS et voix : DescribeRegistrationTypeDefinitions
	SMS et voix : DescribeRegistrationVersions
	SMS et voix : DescribeRegistrations
	SMS et voix : DescribeSenderIds
	SMS et voix : DescribeSpendLimits
	SMS et voix : DescribeVerifiedDestinationNumbers
	SMS et voix : DisassociateOriginationIdentity
	SMS et voix : DisassociateProtectConfiguration
	SMS et voix : DiscardRegistrationVersion
	SMS et voix : GetConfigurationSetEventDestinations
	SMS et voix : GetProtectConfigurationCountryRuleSet
	SMS et voix : GetResourcePolicy
	SMS et voix : ListConfigurationSets
	SMS et voix : ListPoolOriginationIdentities
	SMS et voix : ListProtectConfigurationRuleSetNumberOverrides
	SMS et voix : ListRegistrationAssociations
	SMS et voix : PutKeyword
	SMS et voix : PutOptedOutNumber
	SMS et voix : PutProtectConfigurationRuleSetNumberOverride

Préfixe de service	Actions
	SMS et voix : PutResourcePolicy SMS et voix : ReleasePhoneNumber SMS et voix : ReleaseSenderId SMS et voix : RequestPhoneNumber SMS et voix : RequestSenderId SMS et voix : SendDestinationNumberVerificationCode SMS et voix : SetAccountDefaultProtectConfiguration SMS et voix : SetDefaultMessageFeedbackEnabled SMS et voix : SetDefaultMessageType SMS et voix : SetDefaultSenderId SMS et voix : SetMediaMessageSpendLimitOverride SMS et voix : SetTextMessageSpendLimitOverride SMS et voix : SetVoiceMessageSpendLimitOverride SMS et voix : SubmitRegistrationVersion SMS et voix : UpdateConfigurationSetEventDestination SMS et voix : UpdateEventDestination SMS et voix : UpdatePhoneNumber SMS et voix : UpdatePool SMS et voix : UpdateProtectConfiguration SMS et voix : UpdateProtectConfigurationCountryRuleSet SMS et voix : UpdateSenderId

Préfixe de service	Actions
snowball	boule de neige : CancelCluster
	boule de neige : CancelJob
	boule de neige : CreateAddress
	boule de neige : CreateCluster
	boule de neige : CreateJob
	boule de neige : CreateLongTermPricing
	boule de neige : CreateReturnShippingLabel
	boule de neige : DescribeAddress
	boule de neige : DescribeAddresses
	boule de neige : DescribeCluster
	boule de neige : DescribeJob
	boule de neige : DescribeReturnShippingLabel
	boule de neige : GetJobManifest
	boule de neige : GetJobUnlockCode
	boule de neige : GetSnowballUsage
	boule de neige : GetSoftwareUpdates
	boule de neige : ListClusterJobs
	boule de neige : ListClusters
	boule de neige : ListCompatibleImages
	boule de neige : ListJobs
	boule de neige : ListLongTermPricing

Préfixe de service	Actions
	boule de neige : ListPickupLocations boule de neige : ListServiceVersions boule de neige : UpdateCluster boule de neige : UpdateJob boule de neige : UpdateJobShipmentState boule de neige : UpdateLongTermPricing
sqs	sqs : AddPermission sqs : CancelMessageMoveTask sqs : CreateQueue sqs : DeleteQueue sqs : PurgeQueue sqs : RemovePermission sqs : SetQueueAttributes

Préfixe de service	Actions
ssm	SMS : AssociateOpsItemRelatedItem
	SMS : CancelCommand
	SMS : CancelMaintenanceWindowExecution
	SMS : CreateActivation
	SMS : CreateAssociation
	SMS : CreateAssociationBatch
	SMS : CreateDocument
	SMS : CreateMaintenanceWindow
	SMS : CreateOpsItem
	SMS : CreateOpsMetadata
	SMS : CreatePatchBaseline
	SMS : CreateResourceDataSync
	SMS : DeleteActivation
	SMS : DeleteAssociation
	SMS : DeleteCloudConnector
	SMS : DeleteDocument
	SMS : DeleteInventory
	SMS : DeleteMaintenanceWindow
	SMS : DeleteOpsItem
	SMS : DeleteOpsMetadata
	SMS : DeleteParameter

Préfixe de service	Actions
	SMS : DeleteParameters
	SMS : DeletePatchBaseline
	SMS : DeleteResourceDataSync
	SMS : DeleteResourcePolicy
	SMS : DeregisterManagedInstance
	SMS : DeregisterPatchBaselineForPatchGroup
	SMS : DeregisterTargetFromMaintenanceWindow
	SMS : DeregisterTaskFromMaintenanceWindow
	SMS : DescribeActivations
	SMS : DescribeAssociation
	SMS : DescribeAssociationExecutionTargets
	SMS : DescribeAssociationExecutions
	SMS : DescribeAutomationExecutions
	SMS : DescribeAutomationStepExecutions
	SMS : DescribeAvailablePatches
	SMS : DescribeDocument
	SMS : DescribeDocumentParameters
	SMS : DescribeDocumentPermission
	SMS : DescribeEffectiveInstanceAssociations
	SMS : DescribeEffectivePatchesForPatchBaseline
	SMS : DescribeInstanceAssociationsStatus

Préfixe de service	Actions
	SMS : DescribeInstanceInformation
	SMS : DescribeInstancePatchStates
	SMS : DescribeInstancePatchStatesForPatchGroup
	SMS : DescribeInstancePatches
	SMS : DescribeInstanceProperties
	SMS : DescribeInventoryDeletions
	SMS : DescribeMaintenanceWindowExecutionTaskInvocations
	SMS : DescribeMaintenanceWindowExecutionTasks
	SMS : DescribeMaintenanceWindowExecutions
	SMS : DescribeMaintenanceWindowSchedule
	SMS : DescribeMaintenanceWindowTargets
	SMS : DescribeMaintenanceWindowTasks
	SMS : DescribeMaintenanceWindows
	SMS : DescribeMaintenanceWindowsForTarget
	SMS : DescribeOpsItems
	SMS : DescribeParameters
	SMS : DescribePatchBaselines
	SMS : DescribePatchGroupState
	SMS : DescribePatchGroups
	SMS : DescribePatchProperties
	SMS : DescribeSessions

Préfixe de service	Actions
	SMS : DisassociateOpsItemRelatedItem
	SMS : GetAccessToken
	SMS : GetAutomationExecution
	SMS : GetCalendarState
	SMS : GetCloudConnector
	SMS : GetCommandInvocation
	SMS : GetConnectionStatus
	SMS : GetDefaultPatchBaseline
	SMS : GetDeployablePatchSnapshotForInstance
	SMS : GetDocument
	SMS : GetExecutionPreview
	SMS : GetInventory
	SMS : GetInventorySchema
	SMS : GetMaintenanceWindow
	SMS : GetMaintenanceWindowExecution
	SMS : GetMaintenanceWindowExecutionTask
	SMS : GetMaintenanceWindowExecutionTaskInvocation
	SMS : GetMaintenanceWindowTask
	SMS : GetOpsItem
	SMS : GetOpsMetadata
	SMS : GetOpsSummary

Préfixe de service	Actions
	SMS : GetParameter
	SMS : GetParameterHistory
	SMS : GetParameters
	SMS : GetParametersByPath
	SMS : GetPatchBaseline
	SMS : GetPatchBaselineForPatchGroup
	SMS : GetResourcePolicies
	SMS : GetServiceSetting
	SMS : LabelParameterVersion
	SMS : ListAssociationVersions
	SMS : ListAssociations
	SMS : ListCloudConnectors
	SMS : ListCommandInvocations
	SMS : ListCommands
	SMS : ListComplianceItems
	SMS : ListComplianceSummaries
	SMS : ListDocumentMetadataHistory
	SMS : ListDocumentVersions
	SMS : ListDocuments
	SMS : ListInstanceAssociations
	SMS : ListInventoryEntries

Préfixe de service	Actions
	SMS : ListNodes
	SMS : ListNodesSummary
	SMS : ListOpsItemEvents
	SMS : ListOpsItemRelatedItems
	SMS : ListOpsMetadata
	SMS : ListResourceComplianceSummaries
	SMS : ListResourceDataSync
	SMS : ModifyDocumentPermission
	SMS : PutComplianceItems
	SMS : PutInventory
	SMS : PutParameter
	SMS : PutResourcePolicy
	SMS : RegisterDefaultPatchBaseline
	SMS : RegisterManagedInstance
	SMS : RegisterPatchBaselineForPatchGroup
	SMS : RegisterTargetWithMaintenanceWindow
	SMS : RegisterTaskWithMaintenanceWindow
	SMS : ResetServiceSetting
	SMS : ResumeSession
	SMS : SendAutomationSignal
	SMS : SendCommand

Préfixe de service	Actions
	SMS : StartAssociationsOnce
	SMS : StartAutomationExecution
	SMS : StartChangeRequestExecution
	SMS : StartSession
	SMS : StopAutomationExecution
	SMS : TerminateSession
	SMS : UnlabelParameterVersion
	SMS : UpdateAssociation
	SMS : UpdateAssociationStatus
	SMS : UpdateCloudConnector
	SMS : UpdateDocument
	SMS : UpdateDocumentDefaultVersion
	SMS : UpdateDocumentMetadata
	SMS : UpdateInstanceInformation
	SMS : UpdateMaintenanceWindow
	SMS : UpdateMaintenanceWindowTarget
	SMS : UpdateMaintenanceWindowTask
	SMS : UpdateManagedInstanceRole
	SMS : UpdateOpsItem
	SMS : UpdateOpsMetadata
	SMS : UpdatePatchBaseline

Préfixe de service	Actions
	SMS : UpdateResourceDataSync SMS : UpdateServiceSetting SMS : ValidateCloudConnector

Préfixe de service	Actions
ssm-incidents	Incidents SMS : BatchGetIncidentFindings
	Incidents SMS : CreateReplicationSet
	Incidents SMS : CreateResponsePlan
	Incidents SMS : CreateTimelineEvent
	Incidents SMS : DeleteIncidentRecord
	Incidents SMS : DeleteReplicationSet
	Incidents SMS : DeleteResourcePolicy
	Incidents SMS : DeleteResponsePlan
	Incidents SMS : DeleteTimelineEvent
	Incidents SMS : GetIncidentRecord
	Incidents SMS : GetReplicationSet
	Incidents SMS : GetResourcePolicies
	Incidents SMS : GetResponsePlan
	Incidents SMS : GetTimelineEvent
	Incidents SMS : ListIncidentFindings
	Incidents SMS : ListIncidentRecords
	Incidents SMS : ListRelatedItems
	Incidents SMS : ListReplicationSets
	Incidents SMS : ListResponsePlans
	Incidents SMS : ListTimelineEvents
	Incidents SMS : PutResourcePolicy

Préfixe de service	Actions
	Incidents SMS : StartIncident Incidents SMS : UpdateDeletionProtection Incidents SMS : UpdateIncidentRecord Incidents SMS : UpdateRelatedItems Incidents SMS : UpdateReplicationSet Incidents SMS : UpdateResponsePlan Incidents SMS : UpdateTimelineEvent

Préfixe de service	Actions
ssm-sap	SSM-SAP : BackupDatabase
	SSM-SAP : DeleteResourcePermission
	SSM-SAP : DeregisterApplication
	SSM-SAP : GetApplication
	SSM-SAP : GetComponent
	SSM-SAP : GetConfigurationCheckOperation
	SSM-SAP : GetDatabase
	SSM-SAP : GetOperation
	SSM-SAP : GetResourcePermission
	SSM-SAP : ListApplications
	SSM-SAP : ListComponents
	SSM-SAP : ListConfigurationCheckDefinitions
	SSM-SAP : ListConfigurationCheckOperations
	SSM-SAP : ListDatabases
	SSM-SAP : ListOperationEvents
	SSM-SAP : ListOperations
	SSM-SAP : ListSubCheckResults
	SSM-SAP : ListSubCheckRuleResults
	SSM-SAP : PutResourcePermission
	SSM-SAP : RegisterApplication
	SSM-SAP : RestoreDatabase

Préfixe de service	Actions
	SSM-SAP : StartApplication
	SSM-SAP : StartApplicationRefresh
	SSM-SAP : StartConfigurationChecks
	SSM-SAP : StopApplication
	SSM-SAP : UpdateApplicationSettings
	SSM-SAP : UpdateHANABackupSettings

Préfixe de service	Actions
states	états : CreateActivity états : CreateStateMachine états : CreateStateMachineAlias états : DeleteActivity états : DeleteStateMachine états : DeleteStateMachineAlias états : DeleteStateMachineVersion états : DescribeActivity états : DescribeExecution états : DescribeMapRun états : DescribeStateMachine états : DescribeStateMachineAlias états : DescribeStateMachineForExecution états : GetExecutionHistory états : ListActivities états : ListExecutions états : ListMapRuns états : ListStateMachineAliases états : ListStateMachineVersions états : ListStateMachines états : SendTaskFailure

Préfixe de service	Actions
	états : SendTaskHeartbeat
	états : SendTaskSuccess
	états : StartExecution
	états : StopExecution
	états : UpdateMapRun
	états : UpdateStateMachine
	états : UpdateStateMachineAlias
	états : ValidateStateMachineDefinition
sts	sets : AssumeRole
	sets : AssumeRoleWithSAML
	sets : AssumeRoleWithWebIdentity
	sets : DecodeAuthorizationMessage
	sets : GetAccessKeyInfo
	sets : GetCallerIdentity
	sets : GetFederationToken
	sets : GetSessionToken
	sets : GetWebIdentityToken

Préfixe de service	Actions
swf	swf : DeleteActivityType swf : DeleteWorkflowType swf : DeprecateActivityType swf : DeprecateDomain swf : DeprecateWorkflowType swf : DescribeActivityType swf : DescribeDomain swf : DescribeWorkflowType swf : ListActivityTypes swf : ListDomains swf : ListWorkflowTypes swf : RegisterActivityType swf : RegisterDomain swf : RegisterWorkflowType swf : UndeprecateActivityType swf : UndeprecateDomain swf : UndeprecateWorkflowType

Préfixe de service	Actions
synthetics	synthétiques : AssociateResource
	synthétiques : CreateCanary
	synthétiques : CreateGroup
	synthétiques : DeleteCanary
	synthétiques : DeleteGroup
	synthétiques : DescribeCanaries
	synthétiques : DescribeCanariesLastRun
	synthétiques : DescribeRuntimeVersions
	synthétiques : DisassociateResource
	synthétiques : GetCanary
	synthétiques : GetCanaryRuns
	synthétiques : GetGroup
	synthétiques : ListAssociatedGroups
	synthétiques : ListGroupResources
	synthétiques : ListGroups
	synthétiques : StartCanary
	synthétiques : StartCanaryDryRun
	synthétiques : StopCanary
	synthétiques : UpdateCanary

Préfixe de service	Actions
balise	étiquette : DescribeReportCreation étiquette : GetComplianceSummary étiquette : GetResources étiquette : StartReportCreation

Préfixe de service	Actions
textract	extrait : AnalyzeDocument
	extrait : AnalyzeExpense
	textract:AnalyzeID
	extrait : CreateAdapter
	extrait : CreateAdapterVersion
	extrait : DeleteAdapter
	extrait : DeleteAdapterVersion
	extrait : DetectDocumentText
	extrait : GetAdapter
	extrait : GetAdapterVersion
	extrait : GetDocumentAnalysis
	extrait : GetDocumentTextDetection
	extrait : GetExpenseAnalysis
	extrait : GetLendingAnalysis
	extrait : GetLendingAnalysisSummary
	extrait : ListAdapterVersions
	extrait : ListAdapters
	extrait : StartDocumentAnalysis
	extrait : StartDocumentTextDetection
	extrait : StartExpenseAnalysis
	extrait : StartLendingAnalysis

Préfixe de service	Actions
	extrait : UpdateAdapter

Préfixe de service	Actions
timestream	diffusion chronologique : CancelQuery diffusion chronologique : CreateDatabase diffusion chronologique : CreateScheduledQuery diffusion chronologique : CreateTable diffusion chronologique : DeleteDatabase diffusion chronologique : DeleteScheduledQuery diffusion temporelle : DeleteTable diffusion temporelle : DescribeAccountSettings diffusion temporelle : DescribeDatabase diffusion temporelle : DescribeScheduledQuery diffusion temporelle : DescribeTable diffusion temporelle : ExecuteScheduledQuery diffusion temporelle : ListBatchLoadTasks diffusion temporelle : ListDatabases diffusion temporelle : ListScheduledQueries diffusion temporelle : ListTables diffusion temporelle : PrepareQuery diffusion temporelle : UpdateAccountSettings diffusion temporelle : UpdateDatabase diffusion temporelle : UpdateScheduledQuery diffusion temporelle : UpdateTable

Préfixe de service	Actions
tnb	tnb : CancelSolNetworkOperation tnb : CreateSolFunctionPackage tnb : CreateSolNetworkInstance tnb : CreateSolNetworkPackage tnb : DeleteSolFunctionPackage tnb : DeleteSolNetworkInstance tnb : DeleteSolNetworkPackage tnb : GetSolFunctionInstance tnb : GetSolFunctionPackage tnb : GetSolFunctionPackageContent tnb : GetSolFunctionPackageDescriptor tnb : GetSolNetworkInstance tnb : GetSolNetworkOperation tnb : GetSolNetworkPackage tnb : GetSolNetworkPackageContent tnb : GetSolNetworkPackageDescriptor tnb : InstantiateSolNetworkInstance tnb : ListSolFunctionInstances tnb : ListSolFunctionPackages tnb : ListSolNetworkInstances tnb : ListSolNetworkOperations

Préfixe de service	Actions
	tnb : ListSolNetworkPackages tnb : PutSolFunctionPackageContent tnb : PutSolNetworkPackageContent tnb : TerminateSolNetworkInstance tnb : UpdateSolFunctionPackage tnb : UpdateSolNetworkInstance tnb : UpdateSolNetworkPackage tnb : ValidateSolFunctionPackageContent tnb : ValidateSolNetworkPackageContent

Préfixe de service	Actions
transcribe	transcrire : CreateCallAnalyticsCategory transcrire : CreateLanguageModel transcrire : CreateMedicalVocabulary transcrire : CreateVocabulary transcrire : CreateVocabularyFilter transcrire : DeleteCallAnalyticsCategory transcrire : DeleteCallAnalyticsJob transcrire : DeleteLanguageModel transcrire : DeleteMedicalScribeJob transcrire : DeleteMedicalTranscriptionJob transcrire : DeleteMedicalVocabulary transcrire : DeleteTranscriptionJob transcrire : DeleteVocabulary transcrire : DeleteVocabularyFilter transcrire : DescribeLanguageModel transcrire : GetCallAnalyticsCategory transcrire : GetCallAnalyticsJob transcrire : GetMedicalScribeJob transcrire : GetMedicalTranscriptionJob transcrire : GetMedicalVocabulary transcrire : GetTranscriptionJob

Préfixe de service	Actions
	transcrire : GetVocabulary transcrire : GetVocabularyFilter transcrire : ListCallAnalyticsCategories transcrire : ListCallAnalyticsJobs transcrire : ListLanguageModels transcrire : ListMedicalScribeJobs transcrire : ListMedicalTranscriptionJobs transcrire : ListMedicalVocabularies transcrire : ListTranscriptionJobs transcrire : ListVocabularies transcrire : ListVocabularyFilters transcrire : StartCallAnalyticsJob transcrire : StartCallAnalyticsStreamTranscription transcrire : StartCallAnalyticsStreamTranscriptionWebSocket transcrire : StartMedicalScribeJob transcrire : StartMedicalStreamTranscription transcrire : StartMedicalStreamTranscriptionWebSocket transcrire : StartMedicalTranscriptionJob transcrire : StartStreamTranscription transcrire : StartStreamTranscriptionWebSocket transcrire : StartTranscriptionJob

Préfixe de service	Actions
	transcrire : UpdateCallAnalyticsCategory transcrire : UpdateMedicalVocabulary transcrire : UpdateVocabulary transcrire : UpdateVocabularyFilter

Préfixe de service	Actions
transfert	transfert : CreateAccess
	transfert : CreateAgreement
	transfert : CreateConnector
	transfert : CreateProfile
	transfert : CreateServer
	transfert : CreateUser
	transfert : CreateWebApp
	transfert : CreateWorkflow
	transfert : DeleteAccess
	transfert : DeleteAgreement
	transfert : DeleteCertificate
	transfert : DeleteConnector
	transfert : DeleteHostKey
	transfert : DeleteProfile
	transfert : DeleteServer
	transfert : DeleteSshPublicKey
	transfert : DeleteUser
	transfert : DeleteWebApp
	transfert : DeleteWebAppCustomization
	transfert : DeleteWorkflow
	transfert : DescribeAccess

Préfixe de service	Actions
	transfert : DescribeAgreement
	transfert : DescribeCertificate
	transfert : DescribeConnector
	transfert : DescribeExecution
	transfert : DescribeHostKey
	transfert : DescribeProfile
	transfert : DescribeSecurityPolicy
	transfert : DescribeServer
	transfert : DescribeUser
	transfert : DescribeWebApp
	transfert : DescribeWebAppCustomization
	transfert : DescribeWorkflow
	transfert : ImportCertificate
	transfert : ImportHostKey
	transfert : ImportSshPublicKey
	transfert : ListAccesses
	transfert : ListCertificates
	transfert : ListConnectors
	transfert : ListExecutions
	transfert : ListFileTransferResults
	transfert : ListHostKeys

Préfixe de service	Actions
	transfert : ListProfiles
	transfert : ListSecurityPolicies
	transfert : ListServers
	transfert : ListUsers
	transfert : ListWebApps
	transfert : ListWorkflows
	transfert : SendWorkflowStepState
	transfert : StartDirectoryListing
	transfert : StartFileTransfer
	transfert : StartRemoteDelete
	transfert : StartRemoteMove
	transfert : StartServer
	transfert : StopServer
	transfert : TestConnection
	transfert : TestIdentityProvider
	transfert : UpdateAccess
	transfert : UpdateAgreement
	transfert : UpdateCertificate
	transfert : UpdateConnector
	transfert : UpdateHostKey
	transfert : UpdateProfile

Préfixe de service	Actions
	transfert : UpdateServer transfert : UpdateUser transfert : UpdateWebApp transfert : UpdateWebAppCustomization
translate	traduire : CreateParallelData traduire : DeleteParallelData traduire : DeleteTerminology traduire : DescribeTextTranslationJob traduire : GetParallelData traduire : GetTerminology traduire : ImportTerminology traduire : ListLanguages traduire : ListParallelData traduire : ListTerminologies traduire : ListTextTranslationJobs traduire : StartTextTranslationJob traduire : StopTextTranslationJob traduire : TranslateDocument traduire : TranslateText traduire : UpdateParallelData

Préfixe de service	Actions
voiceid	identifiant vocal : AssociateFraudster
	identifiant vocal : CreateDomain
	identifiant vocal : CreateWatchlist
	identifiant vocal : DeleteDomain
	identifiant vocal : DeleteFraudster
	identifiant vocal : DeleteSpeaker
	identifiant vocal : DeleteWatchlist
	identifiant vocal : DescribeDomain
	identifiant vocal : DescribeFraudster
	identifiant vocal : DescribeFraudsterRegistrationJob
	identifiant vocal : DescribeSpeaker
	identifiant vocal : DescribeSpeakerEnrollmentJob
	identifiant vocal : DescribeWatchlist
	identifiant vocal : DisassociateFraudster
	identifiant vocal : EvaluateSession
	identifiant vocal : ListDomains
	identifiant vocal : ListFraudsterRegistrationJobs
	identifiant vocal : ListFraudsters
	identifiant vocal : ListSpeakerEnrollmentJobs
	identifiant vocal : ListSpeakers
	identifiant vocal : ListWatchlists

Préfixe de service	Actions
	identifiant vocal : OptOutSpeaker
	identifiant vocal : StartFraudsterRegistrationJob
	identifiant vocal : StartSpeakerEnrollmentJob
	identifiant vocal : UpdateDomain
	identifiant vocal : UpdateWatchlist

Préfixe de service	Actions
vpc-lattice	réseau VPC : CreateAccessLogSubscription réseau VPC : CreateListener réseau VPC : CreateResourceConfiguration réseau VPC : CreateResourceGateway réseau VPC : CreateRule réseau VPC : CreateService réseau VPC : CreateServiceNetwork réseau VPC : CreateServiceNetworkResourceAssociation réseau VPC : CreateServiceNetworkServiceAssociation réseau VPC : CreateServiceNetworkVpcAssociation réseau VPC : CreateTargetGroup réseau VPC : DeleteAccessLogSubscription réseau VPC : DeleteAuthPolicy réseau VPC : DeleteDomainVerification réseau VPC : DeleteListener réseau VPC : DeleteResourceConfiguration réseau VPC : DeleteResourceEndpointAssociation réseau VPC : DeleteResourceGateway réseau VPC : DeleteResourcePolicy réseau VPC : DeleteRule réseau VPC : DeleteService

Préfixe de service	Actions
	réseau VPC : DeleteServiceNetwork réseau VPC : DeleteServiceNetworkResourceAssociation réseau VPC : DeleteServiceNetworkServiceAssociation réseau VPC : DeleteServiceNetworkVpcAssociation réseau VPC : DeleteTargetGroup réseau VPC : DeregisterTargets réseau VPC : GetAccessLogSubscription réseau VPC : GetAuthPolicy réseau VPC : GetDomainVerification réseau VPC : GetListener réseau VPC : GetResourceConfiguration réseau VPC : GetResourceGateway réseau VPC : GetResourcePolicy réseau VPC : GetRule réseau VPC : GetService réseau VPC : GetServiceNetwork réseau VPC : GetServiceNetworkResourceAssociation réseau VPC : GetServiceNetworkServiceAssociation réseau VPC : GetServiceNetworkVpcAssociation réseau VPC : GetTargetGroup réseau VPC : ListAccessLogSubscriptions

Préfixe de service	Actions
	réseau VPC : ListDomainVerifications réseau VPC : ListListeners réseau VPC : ListResourceConfigurations réseau VPC : ListResourceEndpointAssociations réseau VPC : ListResourceGateways réseau VPC : ListRules réseau VPC : ListServiceNetworkResourceAssociations réseau VPC : ListServiceNetworkServiceAssociations réseau VPC : ListServiceNetworkVpcAssociations réseau VPC : ListServiceNetworkVpcEndpointAssociations réseau VPC : ListServiceNetworks réseau VPC : ListServices réseau VPC : ListTargetGroups réseau VPC : ListTargets réseau VPC : PutAuthPolicy réseau VPC : PutResourcePolicy réseau VPC : RegisterTargets réseau VPC : StartDomainVerification réseau VPC : UpdateAccessLogSubscription réseau VPC : UpdateListener réseau VPC : UpdateResourceConfiguration

Préfixe de service	Actions
	réseau VPC : UpdateResourceGateway réseau VPC : UpdateRule réseau VPC : UpdateService réseau VPC : UpdateServiceNetwork réseau VPC : UpdateServiceNetworkVpcAssociation réseau VPC : UpdateTargetGroup

Préfixe de service	Actions
wafv2	wafv2 : ACL AssociateWeb
	wafv2 : CheckCapacity
	wafv2:CreateAPIKey
	wafv2:CreateIPSet
	wafv2 : CreateRegexPatternSet
	wafv2 : CreateRuleGroup
	wafv2 : ACL CreateWeb
	wafv2:DeleteAPIKey
	wafv2 : DeleteFirewallManagerRuleGroups
	wafv2:DeleteIPSet
	wafv2 : DeleteLoggingConfiguration
	wafv2 : DeletePermissionPolicy
	wafv2 : DeleteRegexPatternSet
	wafv2 : DeleteRuleGroup
	wafv2 : ACL DeleteWeb
	wafv2 : DescribeAllManagedProducts
	wafv2 : DescribeManagedProductsByVendor
	wafv2 : DescribeManagedRuleGroup
	wafv2 : ACL DisassociateWeb
	wafv2 : GenerateMobileSdkReleaseUrl
	wafv2 : GetDecryptedAPIKey

Préfixe de service	Actions
	wafv2:GetIPSet
	wafv2 : GetLoggingConfiguration
	wafv2 : GetManagedRuleSet
	wafv2 : GetMobileSdkRelease
	wafv2 : GetRateBasedStatementManagedKeys
	wafv2 : GetRegexPatternSet
	wafv2 : GetRevenueStatistics
	wafv2 : GetRevenueStatisticsSummary
	wafv2 : GetRevenueStatisticsTimeSeries
	wafv2 : GetRuleGroup
	wafv2 : GetSampledRequests
	wafv2 : GetWebACLForResource
	wafv2:ListAPIKeys
	wafv2 : ListAvailableManagedRuleGroupVersions
	wafv2 : ListAvailableManagedRuleGroups
	wafv2:ListIPSets
	wafv2 : ListLoggingConfigurations
	wafv2 : ListManagedRuleSets
	wafv2 : ListMobileSdkReleases
	wafv2 : ListRegexPatternSets
	wafv2 : ListResourcesForWebACL

Préfixe de service	Actions
	wafv2 : ListRuleGroups wafv2 : ListSettlementRecords wafv2 : ListWebACLs wafv2 : PutLoggingConfiguration wafv2 : PutManagedRuleSetVersions wafv2:UpdateIPSet wafv2 : UpdateManagedRuleSetVersionExpiryDate wafv2 : UpdateRegexPatternSet wafv2 : UpdateRuleGroup wafv2 : ACL UpdateWeb

Préfixe de service	Actions
wellarchitected	bien architecturé : AssociateLenses bien architecturé : AssociateProfiles bien architecturé : CreateLensShare bien architecturé : CreateLensVersion bien architecturé : CreateMilestone bien architecturé : CreateProfile bien architecturé : CreateProfileShare bien architecturé : CreateReviewTemplate bien architecturé : CreateWorkload bien architecturé : CreateWorkloadShare bien architecturé : DeleteLens bien architecturé : DeleteLensShare bien architecturé : DeleteProfile bien architecturé : DeleteProfileShare bien architecturé : DeleteReviewTemplate bien architecturé : DeleteTemplateShare bien architecturé : DeleteWorkload bien architecturé : DeleteWorkloadShare bien architecturé : DisassociateLenses bien architecturé : DisassociateProfiles bien architecturé : ExportLens

Préfixe de service	Actions
	bien architecturé : GetAnswer bien architecturé : GetConsolidatedReport bien architecturé : GetGlobalSettings bien architecturé : GetLens bien architecturé : GetLensReview bien architecturé : GetLensReviewReport bien architecturé : GetLensVersionDifference bien architecturé : GetMilestone bien architecturé : GetProfile bien architecturé : GetProfileTemplate bien architecturé : GetReviewTemplate bien architecturé : GetReviewTemplateAnswer bien architecturé : GetReviewTemplateLensReview bien architecturé : GetWorkload bien architecturé : ImportLens bien architecturé : ListAnswers bien architecturé : ListCheckDetails bien architecturé : ListCheckSummaries bien architecturé : ListLensReviewImprovements bien architecturé : ListLensReviews bien architecturé : ListLensShares

Préfixe de service	Actions
	bien architecturé : ListLenses bien architecturé : ListMilestones bien architecturé : ListNotifications bien architecturé : ListProfileNotifications bien architecturé : ListProfileShares bien architecturé : ListProfiles bien architecturé : ListReviewTemplateAnswers bien architecturé : ListReviewTemplates bien architecturé : ListShareInvitations bien architecturé : ListTemplateShares bien architecturé : ListWorkloadShares bien architecturé : ListWorkloads bien architecturé : UpdateAnswer bien architecturé : UpdateGlobalSettings bien conçu : UpdateIntegration bien conçu : UpdateLensReview bien conçu : UpdateProfile bien conçu : UpdateReviewTemplate bien conçu : UpdateReviewTemplateLensReview bien conçu : UpdateShareInvitation bien conçu : UpdateWorkload

Préfixe de service	Actions
	bien conçu : UpdateWorkloadShare bien conçu : UpgradeLensReview bien conçu : UpgradeProfileVersion bien conçu : UpgradeReviewTemplateLensReview

Préfixe de service	Actions
wisdom	sagesse : CreateAssistant
	sagesse : CreateAssistantAssociation
	sagesse : CreateContent
	sagesse : CreateKnowledgeBase
	sagesse : CreateQuickResponse
	sagesse : CreateSession
	sagesse : DeleteAssistant
	sagesse : DeleteAssistantAssociation
	sagesse : DeleteContent
	sagesse : DeleteImportJob
	sagesse : DeleteKnowledgeBase
	sagesse : DeleteQuickResponse
	sagesse : GetAssistant
	sagesse : GetAssistantAssociation
	sagesse : GetContent
	sagesse : GetContentAssociation
	sagesse : GetContentSummary
	sagesse : GetImportJob
	sagesse : GetKnowledgeBase
	sagesse : GetRecommendations
	sagesse : GetSession

Préfixe de service	Actions
	sagesse : ListAssistantAssociations
	sagesse : ListAssistants
	sagesse : ListContentAssociations
	sagesse : ListContents
	sagesse : ListImportJobs
	sagesse : ListKnowledgeBases
	sagesse : ListQuickResponses
	sagesse : NotifyRecommendationsReceived
	sagesse : QueryAssistant
	sagesse : RemoveKnowledgeBaseTemplateUri
	sagesse : SearchContent
	sagesse : SearchQuickResponses
	sagesse : SearchSessions
	sagesse : StartContentUpload
	sagesse : StartImportJob
	sagesse : UpdateContent
	sagesse : UpdateKnowledgeBaseTemplateUri
	sagesse : UpdateQuickResponse
	sagesse : UpdateSession

Préfixe de service	Actions
worklink	lien de travail : AssociateDomain lien de travail : AssociateWebsiteAuthorizationProvider lien de travail : AssociateWebsiteCertificateAuthority lien de travail : CreateFleet lien de travail : DeleteFleet lien de travail : DescribeAuditStreamConfiguration lien de travail : DescribeCompanyNetworkConfiguration lien de travail : DescribeDevice lien de travail : DescribeDevicePolicyConfiguration lien de travail : DescribeDomain lien de travail : DescribeFleetMetadata lien de travail : DescribeIdentityProviderConfiguration lien de travail : DescribeWebsiteCertificateAuthority lien de travail : DisassociateDomain lien de travail : DisassociateWebsiteAuthorizationProvider lien de travail : DisassociateWebsiteCertificateAuthority lien de travail : ListDevices lien de travail : ListDomains lien de travail : ListFleets lien de travail : ListWebsiteAuthorizationProviders lien de travail : ListWebsiteCertificateAuthorities

Préfixe de service	Actions
	lien de travail : RestoreDomainAccess lien de travail : RevokeDomainAccess lien de travail : SignOutUser lien de travail : UpdateAuditStreamConfiguration lien de travail : UpdateCompanyNetworkConfiguration lien de travail : UpdateDevicePolicyConfiguration lien de travail : UpdateDomainMetadata lien de travail : UpdateFleetMetadata lien de travail : UpdateIdentityProviderConfiguration

Préfixe de service	Actions
espaces de travail	espaces de travail : AcceptAccountLinkInvitation
	espaces de travail : AssociateConnectionAlias
	espaces de travail : AssociateIpGroups
	espaces de travail : AssociateWorkspaceApplication
	espaces de travail : CopyWorkspaceImage
	espaces de travail : CreateAccountLinkInvitation
	espaces de travail : CreateConnectClientAddIn
	espaces de travail : CreateConnectionAlias
	espaces de travail : CreateIpGroup
	espaces de travail : CreateStandbyWorkspaces
	espaces de travail : CreateUpdatedWorkspaceImage
	espaces de travail : CreateWorkspaceBundle
	espaces de travail : CreateWorkspaceImage
	espaces de travail : CreateWorkspaces
	espaces de travail : CreateWorkspacesPool
	espaces de travail : DeleteAccountLinkInvitation
	espaces de travail : DeleteClientBranding
	espaces de travail : DeleteConnectClientAddIn
	espaces de travail : DeleteConnectionAlias
	espaces de travail : DeleteIpGroup
	espaces de travail : DeleteWorkspaceBundle

Préfixe de service	Actions
	espaces de travail : DeleteWorkspaceImage
	espaces de travail : DeployWorkspaceApplications
	espaces de travail : DeregisterWorkspaceDirectory
	espaces de travail : DescribeAccount
	espaces de travail : DescribeAccountModifications
	espaces de travail : DescribeApplicationAssociations
	espaces de travail : DescribeApplications
	espaces de travail : DescribeBundleAssociations
	espaces de travail : DescribeClientBranding
	espaces de travail : DescribeClientProperties
	espaces de travail : DescribeConnectClientAddIns
	espaces de travail : DescribeConnectionAliasPermissions
	espaces de travail : DescribeConnectionAliases
	espaces de travail : DescribeCustomWorkspaceImageImport
	espaces de travail : DescribeImageAssociations
	espaces de travail : DescribeIpGroups
	espaces de travail : DescribeWorkspaceAssociations
	espaces de travail : DescribeWorkspaceBundles
	espaces de travail : DescribeWorkspaceDirectories
	espaces de travail : DescribeWorkspaceImagePermissions
	espaces de travail : DescribeWorkspaceSnapshots

Préfixe de service	Actions
	espaces de travail : DescribeWorkspaces
	espaces de travail : DescribeWorkspacesConnectionStatus
	espaces de travail : DescribeWorkspacesPoolSessions
	espaces de travail : DescribeWorkspacesPools
	espaces de travail : DisassociateConnectionAlias
	espaces de travail : DisassociateIpGroups
	espaces de travail : DisassociateWorkspaceApplication
	espaces de travail : GetAccountLink
	espaces de travail : ImportClientBranding
	espaces de travail : ImportWorkspaceImage
	espaces de travail : ListAccountLinks
	espaces de travail : ListAvailableManagementCidrRanges
	espaces de travail : MigrateWorkspace
	espaces de travail : ModifyAccount
	espaces de travail : ModifyCertificateBasedAuthProperties
	espaces de travail : ModifyClientProperties
	espaces de travail : ModifyEndpointEncryptionMode
	espaces de travail : ModifySamlProperties
	espaces de travail : ModifySelfservicePermissions
	espaces de travail : ModifyStreamingProperties
	espaces de travail : ModifyWorkspaceAccessProperties

Préfixe de service	Actions
	espaces de travail : ModifyWorkspaceCreationProperties
	espaces de travail : ModifyWorkspaceProperties
	espaces de travail : ModifyWorkspaceState
	espaces de travail : RebootWorkspaces
	espaces de travail : RebuildWorkspaces
	espaces de travail : RegisterWorkspaceDirectory
	espaces de travail : RejectAccountLinkInvitation
	espaces de travail : RestoreWorkspace
	espaces de travail : StartWorkspaces
	espaces de travail : StartWorkspacesPool
	espaces de travail : StopWorkspaces
	espaces de travail : StopWorkspacesPool
	espaces de travail : TerminateWorkspaces
	espaces de travail : TerminateWorkspacesPool
	espaces de travail : TerminateWorkspacesPoolSession
	espaces de travail : UpdateConnectClientAddIn
	espaces de travail : UpdateConnectionAliasPermission
	espaces de travail : UpdateWorkspaceBundle
	espaces de travail : UpdateWorkspaceImagePermission
	espaces de travail : UpdateWorkspacesPool

Préfixe de service	Actions
xray	radiographie : CreateGroup radiographie : CreateSamplingRule radiographie : DeleteGroup radiographie : DeleteResourcePolicy radiographie : DeleteSamplingRule radiographie : GetEncryptionConfig radiographie : GetGroup radiographie : GetGroups radiographie : GetInsight radiographie : GetInsightEvents radiographie : GetInsightImpactGraph radiographie : GetInsightSummaries radiographie : GetSamplingRules radiographie : ListResourcePolicies radiographie : PutEncryptionConfig radiographie : PutResourcePolicy radiographie : UpdateGroup radiographie : UpdateSamplingRule

Quotas de l'IAM Access Analyzer

IAM Access Analyzer a les quotas suivants :

Ressource	Quota par défaut	Quota maximal
Nombre maximum d’analyseurs au niveau du compte par type d’analyseur et par région Compte AWS	1	1
Nombre maximum d'analyseurs d'accès externes ou inutilisés au niveau de l'organisation par type d'analyseur et par région Compte AWS	5	20[1]
Nombre maximum d’analyseurs d’accès internes au niveau de l’organisation par organisation et par région AWS	1	1
Règles d'archivage maximum par analyseur	100 Chaque règle d'archivage peut comporter jusqu'à 20 valeurs par critère.	1 000[1]
Nombre maximum de prévisualisations d'accès par analyseur et par heure	1 000	1 000
AWS CloudTrail fichiers journaux traités par génération de politiques	100 000	100 000
Générations de politique simultanées	1	1

Ressource	Quota par défaut	Quota maximal
Taille des AWS CloudTrail données de génération de politiques	25 Go	25 Go
Échelle de AWS CloudTrail temps de génération des politiques	90 jours	90 jours
Générations de politiques par jour	Afrique (Le Cap) : 5 Asie-Pacifique (Hong Kong) : 5 Asie-Pacifique (Jakarta) : 5 Europe (Milan) : 5 Moyen-Orient (Bahreïn) : 5 Toutes les autres régions prises en charge : 50 **Note** Les demandes de génération de politique annulées s'appliquent au quota quotidien.	Afrique (Le Cap) : 5 Asie-Pacifique (Hong Kong) : 5 Asie-Pacifique (Jakarta) : 5 Europe (Milan) : 5 Moyen-Orient (Bahreïn) : 5 Toutes les autres régions prises en charge : 50

[1]Certains quotas sont configurables par le client via l'option Service Quotas (service des quotas).

Résolutions des problèmes IAM

Utilisez les informations ici pour vous aider à diagnostiquer et résoudre les problèmes courants lorsque vous utilisez Gestion des identités et des accès AWS (IAM).

Problèmes

- Je n'arrive pas à me connecter à mon AWS compte
- J'ai perdu mes clés d'accès
- Les variables de la politique ne fonctionnent pas
- Les modifications que j'apporte ne sont pas toujours visibles immédiatement
- Je ne suis pas autorisé à effectuer : iam : DeleteVirtualMFADevice
- Comment créer des utilisateurs IAM en toute sécurité ?
- Ressources supplémentaires
- Résolution des problèmes liés aux messages d'erreur d'accès rejeté
- Résoudre les messages d'erreur relatifs à l'accès refusé avec l'ID d'autorisation (version préliminaire)
- Résolution de problèmes relatifs à l'utilisateur racine
- Dépannage de politiques IAM
- Résolution des problèmes liés aux clés d'accès et aux clés de sécurité FIDO
- Résolution des problèmes liés aux rôles IAM
- Dépannage d'IAM et Amazon EC2
- Résolution des problèmes liés à IAM et Amazon S3
- Résolution des problèmes liés à la fédération SAML avec IAM

Je n'arrive pas à me connecter à mon AWS compte

Vérifiez que vous disposez des informations d'identification correctes et que vous utilisez la bonne méthode pour vous connecter. Pour plus d'informations, consultez Résolution des problèmes de connexion dans le Guide de l'utilisateur Connexion à AWS .

J'ai perdu mes clés d'accès

Les clés d'accès se composent de deux parties :

- Identificateur de clé d'accès. Cet élément n'est pas secret et peut être affiché dans la console IAM partout où les clés d'accès sont répertoriées, comme sur la page de récapitulatif de l'utilisateur.
- Clé d'accès secrète. Cette partie est fournie lorsque vous créez initialement la paire de clés d'accès. Tout comme un mot de passe, elle ne peut pas être récupérée ultérieurement. Si vous perdez votre clé d'accès secrète, vous devez créer une nouvelle paire de clés d'accès. Si vous disposez déjà du nombre maximum de clés d'accès, vous devez supprimer une paire existante avant de pouvoir en créer une autre.

Si vous perdez votre clé d'accès secrète, vous devez supprimer la clé d'accès et en créer une nouvelle. Pour plus d'informations, consultez Mise à jour des clés d'accès.

Les variables de la politique ne fonctionnent pas

Si vos variables politiques ne fonctionnent pas, l'une des erreurs suivantes s'est produite :

La date est incorrecte dans l'élément Politique de version.

Vérifiez que toutes les politiques incluant des variables contiennent le numéro de version suivant : `"Version": "2012-10-17"`. Sans le numéro de version approprié, les variables ne sont pas remplacées lors de l'évaluation. Au contraire, les variables sont évaluées littéralement. Les politiques n'incluant pas de variables continuent de fonctionner lorsque vous incluez le numéro de version le plus récent.

Un élément de politique `Version` est différent d'une version de politique. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. Une version de politique est créée lorsque vous apportez des modifications à une politique gérée par le client dans IAM. La politique modifiée ne remplace pas la politique existante. À la place, IAM crée une nouvelle version de la politique gérée. Pour en savoir plus sur l'élément de politique `Version`, consultez Éléments de politique JSON IAM : Version. Pour en savoir plus sur les versions de politiques, consultez the section called "Gestion des versions des politiques IAM".

Les caractères de la variable ne respectent pas la casse.

Vérifiez que vos variables de politique sont dans la casse correcte. Pour plus d'informations, veuillez consulter Éléments des politiques IAM : variables et balises.

Les modifications que j'apporte ne sont pas toujours visibles immédiatement

En tant que service auquel on accède avec des ordinateurs situés dans des centres de données du monde entier, IAM utilise un modèle d'informatique distribuée appelé cohérence éventuelle. Toutes les modifications que vous apportez à IAM (ou à d'autres AWS services), y compris les balises de contrôle d'accès basé sur les attributs (ABAC), mettent du temps à être visibles sur tous les points de terminaison possibles. Certains retards résultent du temps nécessaire pour envoyer les données d'un serveur à l'autre, d'une zone de réplication à l'autre et d'une région à l'autre. IAM utilise également la mise en cache pour améliorer les performances mais, dans certains cas, cela peut ralentir le processus. La modification peut ne pas être visible tant que les données mises en cache précédemment n'arrivent pas à expiration.

Vous devez concevoir vos applications globales de sorte qu'elles tiennent compte de ces retards potentiels. Assurez-vous qu'elles fonctionnent comme prévu, même lorsqu'une modification effectuée à un emplacement n'est pas visible instantanément à un autre. Les modifications peuvent être la création ou la mise à jour d'utilisateurs, de groupes, de rôles ou de politiques. Nous vous recommandons de ne pas inclure ce type de modifications IAM dans les chemins de code critique et haute disponibilité de votre application. Au lieu de cela, procédez aux modifications IAM dans une routine d'initialisation ou d'installation distincte que vous exécutez moins souvent. Veillez également à vérifier que les modifications ont été propagées avant que les processus de production en dépendent.

Pour plus d'informations sur la manière dont certains autres AWS services sont affectés par cette situation, consultez les ressources suivantes :

- Amazon DynamoDB : Cohérence en lecture dans le Guide du développeur DynamoDB et Cohérence en lecture dans le Guide du développeur Amazon DynamoDB
- Amazon EC2 : Cohérence éventuelle EC2 dans la Référence d'API Amazon EC2
- Amazon EMR :Assurer la cohérence lors de l'utilisation d'Amazon S3 et Amazon EMR pour les flux de travail ETL dans le blog AWS sur les big data
- Amazon Redshift :gestion de la cohérence des données dans le guide du développeur de la base de données Amazon Redshift
- Amazon S3 : modèle de cohérence de données Amazon S3 dans le guide d'utilisateur du service de stockage simple d'Amazon

Je ne suis pas autorisé à effectuer : iam : DeleteVirtualMFADevice

L'erreur suivante peut s'afficher lorsque vous tentez d'attribuer ou de supprimer un dispositif MFA virtuel pour vous-même ou d'autres personnes :

```
User: arn:aws:iam::123456789012:user/Diego is not authorized to
 perform: iam:DeleteVirtualMFADevice on resource: arn:aws:iam::123456789012:mfa/Diego
 with an explicit deny
```

Elle peut se produire si quelqu'un a commencé à assigner un dispositif MFA virtuel à un utilisateur dans la console IAM et a annulé le processus. Cela crée un dispositif MFA virtuel pour l'utilisateur dans IAM, mais ne l'affecte jamais à l'utilisateur. Supprimez le dispositif MFA virtuel existant avant de pouvoir en créer un nouveau portant le même nom.

Pour résoudre ce problème, l'administrateur ne doit pas modifier les autorisations de politique. L'administrateur doit plutôt utiliser l' AWS API AWS CLI or pour supprimer le périphérique MFA virtuel existant mais non attribué.

Supprimer un dispositif MFA virtuel existant, mais non attribué

1. Affichez les dispositifs MFA virtuels de votre compte.
   - AWS CLI: `aws iam list-virtual-mfa-devices`
   - AWS API : `ListVirtualMFADevices`
2. Dans la réponse, localisez l'ARN du dispositif MFA virtuel pour l'utilisateur que vous essayez de réparer.
3. Supprimez le dispositif MFA virtuel.
   - AWS CLI: `aws iam delete-virtual-mfa-device`
   - AWS API : `DeleteVirtualMFADevice`

Comment créer des utilisateurs IAM en toute sécurité ?

Si certains de vos employés ont besoin d'un accès à AWS, vous pouvez choisir de créer des utilisateurs IAM ou d'utiliser IAM Identity Center pour l'authentification. Si vous utilisez IAM, vous AWS recommande de créer un utilisateur IAM et de communiquer les informations d'identification à l'employé en toute sécurité. Si vous ne vous trouvez pas physiquement à côté de votre employé, utilisez un flux de travail sécurisé pour communiquer les informations d'identification à celui-ci.

Utilisez le flux de travail sécurisé suivant pour créer un utilisateur dans IAM :

1. Créez un utilisateur à l'aide de la Console de gestion AWS. Choisissez d'accorder l' Console de gestion AWS accès à l'aide d'un mot de passe généré. Si nécessaire, cochez la case Les utilisateurs doivent créer un mot de passe à leur prochaine connexion. N'ajoutez pas de politique d'autorisations à l'utilisateur tant que celui-ci n'a pas modifié son mot de passe.
2. Une fois l'utilisateur ajouté, copiez l'URL de connexion, le nom d'utilisateur et le mot de passe du nouvel utilisateur. Pour afficher le mot de passe, choisissez afficher.
3. Envoyez le mot de passe à votre employé à l'aide d'une méthode de communication sécurisée dans votre entreprise, par exemple, par e-mail, chat ou système de tickets. Séparément, fournissez à vos utilisateurs le lien de la console utilisateur IAM et leur nom d'utilisateur. Demandez à l'employé de vous confirmer qu'il peut se connecter correctement avant de lui accorder des autorisations.
4. Une fois que l'employé vous a confirmé cela, ajoutez les autorisations dont il a besoin. Pour des raisons de sécurité, il est recommandé d'ajouter une politique qui exige que l'utilisateur s'authentifie à l'aide de MFA pour gérer ses informations d'identification. Pour un exemple de politique, veuillez consulter AWS : autorise les utilisateurs IAM authentifiés par MFA à gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

Ressources supplémentaires

Les ressources suivantes peuvent vous aider à résoudre les problèmes pendant que vous travaillez avec AWS.

- AWS CloudTrail Guide de l'utilisateur : AWS CloudTrail permet de suivre l'historique des appels d'API effectués AWS et de stocker ces informations dans des fichiers journaux. Vous pouvez ainsi identifier les utilisateurs et les comptes ayant accédé aux ressources de votre compte lors des appels, déterminer les actions demandées, etc. Pour de plus amples informations, veuillez consulter Journalisation IAM et AWS STS Appels d'API avec AWS CloudTrail.
- AWS Centre de connaissances — Trouvez des FAQ et des liens vers d'autres ressources pour vous aider à résoudre les problèmes.
- AWS Centre de support — Bénéficiez d'une assistance technique.
- AWS Centre de support Premium — Bénéficiez d'une assistance technique premium.

Résolution des problèmes liés aux messages d'erreur d'accès rejeté

Les informations suivantes peuvent vous aider à identifier, diagnostiquer et résoudre les erreurs d'accès refusé avec Gestion des identités et des accès AWS. Les erreurs d'accès refusé apparaissent lorsque le refus AWS explicite ou implicite d'une demande d'autorisation est refusé.

- Un refus explicite se produit lorsqu'une politique contient une `Deny` déclaration pour l' AWS action spécifique.
- Un refus implicite se produit lorsqu'il n'y a ni d'instruction `Deny` applicable ni d'instruction `Allow` applicable. Comme une politique IAM refuse un principal IAM par défaut, elle doit autoriser explicitement le principal à effectuer une action. Dans le cas contraire, la politique refuse implicitement l'accès. Pour de plus amples informations, veuillez consulter Différence entre les refus explicites et implicites.

Lorsque vous effectuez une requête à un service ou à une ressource, plusieurs politiques peuvent s'appliquer à la requête. Passez en revue toutes les politiques applicables en plus de la politique spécifiée dans le message d'erreur.

- Si plusieurs politiques du même type refusent une requête, le message d'erreur « accès refusé » ne précise pas le nombre de politiques évaluées.
- Si plusieurs types de politiques refusent une demande d'autorisation, n' AWS incluez qu'un seul de ces types de politiques dans le message d'erreur.

> **Important**
>
> Vous rencontrez des difficultés pour vous connecter à AWS ? Assurez-vous que vous êtes sur la bonne page de connexion AWS pour votre type d'utilisateur. Si vous êtes le Utilisateur racine d'un compte AWS (propriétaire du compte), vous pouvez vous connecter à AWS l'aide des informations d'identification que vous avez configurées lors de la création du Compte AWS. Si vous êtes un utilisateur IAM, votre administrateur de compte peut vous fournir des informations d'identification AWS . Si vous avez besoin d'assistance, n'utilisez pas le lien commentaire indiqué sur cette page. Le formulaire est reçu par l'équipe de AWS documentation, non Support. Sur la page Contactez-nous, sélectionnez Impossible toujours

de vous connecter à votre AWS compte, puis choisissez l'une des options d'assistance disponibles.

Je reçois un « accès refusé » lorsque je fais une demande à un AWS web

- Vérifiez si le message d'erreur inclut le type et le nom de ressource Amazon (ARN) de la politique responsable du refus d'accès. Si tel est le cas, vérifiez les instructions de refus pour l'action dans la politique spécifiée. Si le type de stratégie est fourni mais qu'il n'existe pas d'ARN de stratégie, concentrez-vous sur la résolution des problèmes liés à ce type de stratégie : vérifiez les instructions de refus pour l'action dans les stratégies du type spécifié. Si le message d'erreur n'indique pas le type de politique responsable du refus d'accès, utilisez les autres instructions de cette section pour effectuer un dépannage plus approfondi.
- Vérifiez que vous avez l'autorisation de politique basée sur l'identité d'appeler l'action et la ressource que vous avez demandées. Si des conditions sont définies, vous devez également respecter ces conditions lorsque vous envoyez la demande. Pour plus d'informations sur l'affichage ou la modification de politiques pour un utilisateur, un groupe ou un rôle IAM, veuillez consulter Gestion des politiques IAM.
- Si un Console de gestion AWS message indiquant que vous n'êtes pas autorisé à effectuer une action s'affiche, vous devez contacter votre administrateur pour obtenir de l'aide. Votre administrateur vous a fourni vos informations de connexion ou votre lien de connexion.

  L'exemple d'erreur suivant se produit quand l'utilisateur IAM `mateojackson` tente d'utiliser la console pour afficher des informations détaillées sur une ressource *`my-example-widget`* fictive, mais ne dispose pas des autorisations `widgets:`*`GetWidget`* fictives.

  ```
  User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
   widgets:GetWidget on resource: my-example-widget
  ```

  Dans ce cas, Mateo doit demander à son administrateur de mettre à jour ses politiques pour lui permettre d'accéder à la ressource *`my-example-widget`* à l'aide de l'action `widgets:`*`GetWidget`*.
- Essayez-vous d'accéder à un service prenant en charge des politiques basées sur les ressources, tel que Amazon S3, Amazon SNS ou Amazon SQS ? Si tel est le cas, vérifiez que la politique vous spécifie comme principal et vous accorde l'accès. Si vous effectuez une demande à un service dans votre compte, vos politiques basées sur une identité ou sur les ressources peuvent vous

accorder l'autorisation. Si vous effectuez une demande à un service dans un autre compte, vos politiques basées sur une identité et sur les ressources doivent vous accorder l'autorisation. Pour afficher les services qui prennent en charge les politiques basées sur les ressources, veuillez consulter AWS services qui fonctionnent avec IAM.

- Si votre politique inclut une condition avec une paire clé-valeur, vérifiez-la attentivement. Les exemples incluent la clé de condition `aws:RequestTag/tag-key` globale AWS KMS `kms:EncryptionContext:encryption_context_key`, la clé de condition et la clé de `ResourceTag/tag-key` condition prise en charge par plusieurs services. Assurez-vous que le nom de clé ne correspond pas à plusieurs résultats. Les noms de clé de condition n'étant pas sensibles à la casse, une condition qui vérifie une clé nommée `foo` correspond à `foo`, `Foo`, ou `F00`. Si votre demande comprend plusieurs paires clé-valeur avec des noms de clé qui ne diffèrent que par la casse, votre accès pourrait être inopinément refusé. Pour plus d'informations, veuillez consulter Éléments de la politique JSON IAM : Condition.
- Si vous disposez d'une limite d'autorisations, vérifiez que la politique utilisée pour cette dernière autorise votre demande. Si vos politiques basées sur l'identité permettent la demande, mais pas votre limite d'autorisations, la demande est refusée. Une limite d'autorisations contrôle les autorisations maximales qu'un responsable IAM (utilisateur ou rôle) peut avoir. Resource-based les politiques ne sont pas limitées par des limites d'autorisations. Les limites d'autorisations ne sont pas courantes. Pour plus d'informations sur la manière dont AWS les politiques sont évaluées, consultezLogique d'évaluation de politiques.
- Si vous signez des demandes manuellement (sans utiliser les kits SDK AWS), vérifiez que vous avez correctement signé la demande.
- Si vous utilisez une politique de point de terminaison Amazon VPC et que vous obtenez une erreur d'accès refusé qui n'est pas enregistrée dans AWS CloudTrail, cela peut être dû au fait que le compte propriétaire du point de terminaison d'un VPC est différent du compte appelant ou du compte de rôle cible.

Je reçois un message d'accès refusé lorsque j'effectue une demande avec des informations d'identification de sécurité temporaires

- Tout d'abord, assurez-vous que l'accès ne vous est pas refusé pour une raison non liée à vos informations d'identification temporaires. Pour plus d'informations, veuillez consulter Je reçois un « accès refusé » lorsque je fais une demande à un AWS web.
- Pour vérifier que le service accepte les informations d'identification de sécurité temporaires, veuillez consulter AWS services qui fonctionnent avec IAM.

- Vérifiez que vos demandes sont signées correctement et que la demande est correctement formée. Pour en savoir plus, veuillez consulter la documentation de votre boîte à outils ou Utilisez des informations d'identification temporaires avec AWS resources.
- Vérifiez que vos informations d'identification de sécurité temporaires ne sont pas arrivées à expiration. Pour plus d'informations, veuillez consulter Informations d'identification de sécurité temporaires dans IAM.
- Vérifiez que l'utilisateur ou le rôle IAM dispose des autorisations adéquates. Les autorisations pour les informations d'identification de sécurité temporaires sont dérivées d'un utilisateur ou d'un rôle IAM. Par conséquent, les autorisations sont limitées à celles qui sont accordées au rôle dont vous avez utilisé les informations d'identification temporaires. Pour de plus amples informations sur la définition des autorisations des informations d'identification de sécurité temporaires, veuillez consulter Autorisations affectées aux informations d'identification de sécurité temporaires.
- Si vous avez endossé un rôle, votre session de rôle peut être limitée par les politiques de session. Lorsque vous demandez des informations d'identification de sécurité temporaires par programmation à l'aide de AWS STS, vous pouvez éventuellement transmettre des politiques de session en ligne ou gérées. Les politiques de session sont des politiques avancées que vous transmettez en tant que paramètre lorsque vous créez par programmation une session d'informations d'identification temporaires pour un rôle. Vous pouvez transmettre un seul document de politique de session en ligne JSON à l'aide du paramètre `Policy`. Vous pouvez utiliser le paramètre `PolicyArns` pour spécifier jusqu'à 10 politiques de session gérées. Les autorisations de la session obtenues sont une combinaison des politiques basées sur l'identité du rôle et des politiques de session. Sinon, si votre administrateur ou un programme personnalisé vous fournit des informations d'identification temporaires, ils peuvent avoir inclus une politique de session pour limiter votre accès.
- Si vous êtes un utilisateur principal AWS STS fédéré, votre session peut être limitée par des politiques de session. Vous créez une session utilisateur fédérée en vous connectant en AWS tant qu'utilisateur IAM, puis en demandant un jeton de fédération. Pour de plus amples informations, veuillez consulter Demande d'informations d'identification via un fournisseur d'identité personnalisé. Si vous ou votre broker d'identité avez transmis des politiques de session lorsque vous avez demandé un jeton de session, votre session est limitée par ces politiques. Les autorisations de la session obtenues sont une combinaison des politiques basées sur l'identité de l'utilisateur IAM et des politiques de session. Pour de amples informations sur les politiques de session, veuillez consulter Politiques de session.

- Si vous accédez à une ressource disposant d'une politique basée sur une ressource à l'aide d'un rôle, vérifiez que la politique accorde des autorisations au rôle. Par exemple, la politique suivante autorise `MyRole` du compte `111122223333` à accéder à `amzn-s3-demo-bucket`.

  JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Sid": "S3BucketPolicy",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::111122223333:role/MyRole"]},
    "Action": ["s3:PutObject"],
    "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/*"]
  }]
}
```

Exemples de messages d'erreur d'accès refusé

La plupart des messages d'erreur d'accès refusé se présentent sous le format `User` *`user`* `is not authorized to perform` *`action`* `on` *`resource`* `because` *`context`*. Dans cet exemple, *`user`* il s'agit de l'ARN du principal auquel l'accès *`action`* est refusé, de l'action de service refusée par la politique et *`resource`* de l'ARN de la ressource sur laquelle la stratégie agit. Le *`context`* champ fournit un contexte supplémentaire sur le type de politique qui a refusé l'accès. Dans certains cas, il contient également l'ARN de la politique qui a refusé l'accès.

Lorsqu'une politique refuse explicitement l'accès parce qu'elle contient une `Deny` instruction, elle AWS inclut la phrase `with an explicit deny in a` *`type`* `policy` dans le message d'erreur d'accès refusé. Cette phrase peut également spécifier l'ARN de la politique, comme suit :`with an explicit deny in a` *`type`* `policy:` *`policy ARN`*.

Lorsque la politique refuse implicitement l'accès, elle AWS inclut la phrase `because no` *`type`* `policy allows the` *`action`* `action` dans le message d'erreur d'accès refusé.

> **Note**
>
> Certains AWS services ne prennent pas en charge ce format de message d'erreur d'accès refusé. Le contenu des messages d'erreur d'accès refusé peut varier en fonction du service à l'origine de la demande d'autorisation.

Les exemples suivants montrent le format pour différents types de messages d'erreur d'accès refusé.

Accès refusé en raison d'une politique de contrôle des services : refus implicite

1. Vérifiez la présence d'une instruction `Allow` manquante pour l'action dans vos politiques de contrôle des services (SCP). Pour l'exemple suivant, l'action est `codecommit:ListRepositories`.
2. Mettez à jour votre SCP en ajoutant l'instruction `Allow`. Pour plus d'information, consultez Mise à jour d'une stratégie de contrôle de service (SCP) dans le Guide de l'utilisateur AWS Organizations .

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
because no service control policy allows the codecommit:ListRepositories action
```

Accès refusé en raison d'une politique de contrôle des services : refus explicite

1. Si un ARN de stratégie est fourni dans le message d'erreur, recherchez une `Deny` instruction pour l'action dans la politique de contrôle de service (SCP) spécifiée. Dans l'exemple ci-dessous, l'action est`codecommit:ListRepositories`.
2. Si aucun ARN de politique n'est fourni dans le message d'erreur, recherchez une `Deny` instruction concernant l'action dans vos SCP.
3. Mettez à jour votre SCP en supprimant l'instruction `Deny`. Pour plus d'informations, voir Mettre à jour une politique de contrôle de service (SCP) dans le Guide de l'AWS Organizations utilisateur.

Message d'erreur avec un ARN de politique :

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
with an explicit deny in a service control policy:
 arn:aws:organizations::777788889999:policy/o-exampleorgid/service_control_policy/p-
examplepolicyid123
```

Message d'erreur sans ARN de politique :

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
```

```
with an explicit deny in a service control policy
```

Accès refusé en raison d'une politique de contrôle des ressources : refus explicite

1. Si un ARN de politique est fourni dans le message d'erreur, recherchez une `Deny` instruction pour l'action dans la politique de contrôle des ressources (RCP) spécifiée. Dans l'exemple ci-dessous, l'action est`secretsmanager:GetSecretValue`.
2. Si aucun ARN de stratégie n'est fourni dans le message d'erreur, recherchez une `Deny` instruction concernant l'action dans vos RCP.
3. Mettez à jour votre politique de contrôle des ressources en supprimant l'instruction `Deny`. Pour plus d'informations, voir Mettre à jour une politique de contrôle des ressources (RCP) dans le Guide de l'AWS Organizations utilisateur.

Message d'erreur avec un ARN de politique :

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
on resource: arn:aws:secretsmanager:us-east-1:123456789012:secret:*
with an explicit deny in a resource control policy:
 arn:aws:organizations::777788889999:policy/o-exampleorgid/resource_control_policy/p-
examplepolicyid456
```

Message d'erreur sans ARN de politique :

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
on resource: arn:aws:secretsmanager:us-east-1:123456789012:secret:*
with an explicit deny in a resource control policy
```

Accès refusé en raison d'une politique de point de terminaison d'un VPC : refus implicite

1. Vérifiez l'absence d'une instruction `Allow` pour l'action dans vos politiques de point de terminaison de cloud privé virtuel (VPC). Pour l'exemple suivant, l'action est `codecommit:ListRepositories`.
2. Mettez à jour la politique de point de terminaison d'un VPC en ajoutant l'instruction `Allow`. Pour plus d'informations, consultez la rubrique Mise à jour d'une politique de point de terminaison d'un VPC dans le Guide AWS PrivateLink .

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
because no VPC endpoint policy allows the codecommit:ListRepositories action
```

Accès refusé en raison d'une politique de point de terminaison d'un VPC : refus explicite

1. Vérifiez la présence d'une instruction `Deny` explicite pour l'action dans vos politiques de point de terminaison de cloud privé virtuel (VPC). Pour l'exemple suivant, l'action est `codedeploy:ListDeployments`.
2. Mettez à jour la politique de point de terminaison d'un VPC en supprimant l'instruction `Deny`. Pour plus d'informations, veuillez consulter la rubrique Mise à jour d'une politique de point de terminaison d'un VPC dans le Guide AWS PrivateLink .

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in a VPC endpoint policy
```

Accès refusé en raison d'une limite des autorisations : refus implicite

1. Vérifiez l'absence d'une instruction `Allow` pour l'action dans votre limite des autorisations. Pour l'exemple suivant, l'action est `codedeploy:ListDeployments`.
2. Mettez à jour votre limite des autorisations en ajoutant l'instruction `Allow` à votre politique IAM. Pour plus d'informations, consultez Limites d'autorisations pour les entités IAM et Modification de politiques IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
because no permissions boundary allows the codedeploy:ListDeployments action
```

Accès refusé en raison d'une limite des autorisations : refus explicite

1. Si un ARN de politique est fourni dans le message d'erreur, recherchez une `Deny` instruction pour l'action dans la limite d'autorisations spécifiée. Dans l'exemple ci-dessous, l'action est`sagemaker:ListModels`.
2. Si aucun ARN de politique n'est fourni dans le message d'erreur, recherchez une `Deny` instruction pour l'action dans la limite d'autorisations attachée au principal.
3. Mettez à jour votre limite des autorisations en supprimant l'instruction `Deny` de votre politique IAM. Pour plus d'informations, consultez Limites d'autorisations pour les entités IAM et Modification de politiques IAM.

Message d'erreur avec un ARN de politique :

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 sagemaker:ListModels
with an explicit deny in a permissions boundary: arn:aws:iam::123456789012:policy/
DeveloperPermissionBoundary
```

Message d'erreur sans ARN de politique :

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 sagemaker:ListModels
with an explicit deny in a permissions boundary
```

Accès refusé en raison de politiques de session : refus implicite

1. Vérifiez l'absence d'une instruction `Allow` pour l'action dans vos politiques de session. Pour l'exemple suivant, l'action est `codecommit:ListRepositories`.
2. Mettez à jour votre politique de session en ajoutant l'instruction `Allow`. Pour plus d'informations, veuillez consulter les rubriques Politiques de session et Modification de politiques IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
because no session policy allows the codecommit:ListRepositories action
```

Accès refusé en raison de politiques de session : refus explicite

1. Si un ARN de stratégie est fourni dans le message d'erreur, recherchez une `Deny` instruction pour l'action dans la stratégie de session spécifiée. Dans l'exemple ci-dessous, l'action est`codedeploy:ListDeployments`.
2. Si aucun ARN de stratégie n'est fourni dans le message d'erreur, recherchez une `Deny` instruction concernant l'action dans vos politiques de session.
3. Mettez à jour votre politique de session en supprimant l'instruction `Deny`. Pour plus d'informations, veuillez consulter les rubriques Politiques de session et Modification de politiques IAM.

Message d'erreur avec un ARN de politique :

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in a session policy: arn:aws:iam::123456789012:policy/
DeveloperSessionPolicy
```

Message d'erreur sans ARN de politique :

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in a session policy
```

Accès refusé en raison de politiques basées sur les ressources : refus implicite

1. Vérifiez l'absence d'une instruction `Allow` pour l'action dans votre politique basée sur les ressources. Pour l'exemple suivant, l'action est `secretsmanager:GetSecretValue`.
2. Mettez à jour votre politique en ajoutant l'instruction `Allow`. Pour plus d'informations, consultez Resource-based les politiques et Modification de politiques IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
because no resource-based policy allows the secretsmanager:GetSecretValue action
```

Accès refusé en raison de politiques basées sur les ressources : refus explicite

1. Vérifiez la présence d'une instruction `Deny` explicite pour l'action dans votre politique basée sur les ressources. Pour l'exemple suivant, l'action est `secretsmanager:GetSecretValue`.
2. Mettez à jour votre politique en supprimant l'instruction `Deny`. Pour plus d'informations, consultez Resource-based les politiques et Modification de politiques IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
on resource: arn:aws:secretsmanager:us-east-1:123456789012:secret:*
with an explicit deny in a resource-based policy
```

Accès refusé en raison de politiques d'approbation de rôle : refus implicite

1. Vérifiez l'absence d'une instruction `Allow` pour l'action dans votre politique d'approbation de rôle. Pour l'exemple suivant, l'action est `sts:AssumeRole`.
2. Mettez à jour votre politique en ajoutant l'instruction `Allow`. Pour plus d'informations, consultez Resource-based les politiques et Modification de politiques IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform: sts:AssumeRole
because no role trust policy allows the sts:AssumeRole action
```

Accès refusé en raison de politiques d'approbation de rôle : refus explicite

1. Vérifiez la présence d'une instruction `Deny` explicite pour l'action dans votre politique d'approbation de rôle. Pour l'exemple suivant, l'action est `sts:AssumeRole`.
2. Mettez à jour votre politique en supprimant l'instruction `Deny`. Pour plus d'informations, consultez Resource-based les politiques et Modification de politiques IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform: sts:AssumeRole
with an explicit deny in the role trust policy
```

Accès refusé en raison de politiques basées sur l'identité : refus implicite

1. Vérifiez l'absence d'une instruction `Allow` pour l'action dans les politiques basées sur l'identité attachées à l'identité. Pour l'exemple suivant, l'action est `codecommit:ListRepositories` attachée à au rôle HR.
2. Mettez à jour votre politique en ajoutant l'instruction `Allow`. Pour plus d'informations, consultez Identity-based les politiques et Modification de politiques IAM.

```
User: arn:aws:iam::123456789012:role/HR is not authorized to perform:
 codecommit:ListRepositories
because no identity-based policy allows the codecommit:ListRepositories action
```

Accès refusé en raison de politiques basées sur l'identité : refus explicite

1. Si un ARN de stratégie est fourni dans le message d'erreur, recherchez une `Deny` instruction pour l'action dans la stratégie spécifiée. Dans l'exemple ci-dessous, l'action est`codedeploy:ListDeployments`.
2. Si aucun ARN de stratégie n'est fourni dans le message d'erreur, recherchez une `Deny` instruction pour l'action dans les stratégies basées sur l'identité associées à l'identité.
3. Mettez à jour votre politique en supprimant l'instruction `Deny`. Pour plus d'informations, consultez Identity-based les politiques et Modification de politiques IAM.

Message d'erreur avec un ARN de politique :

```
User: arn:aws:iam::123456789012:role/HR is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in an identity-based policy: arn:aws:iam::123456789012:policy/
HRAccessPolicy
```

Message d'erreur sans ARN de politique :

```
User: arn:aws:iam::123456789012:role/HR is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in an identity-based policy
```

Résoudre les messages d'erreur relatifs à l'accès refusé avec l'ID d'autorisation (version préliminaire)

> **Important**
>
> L'utilitaire de résolution des problèmes d'accès est en version préliminaire publique.

Vue d'ensemble

En cas AWS de refus d'une demande pour une API prise en charge, la réponse d'erreur d'accès refusé inclut généralement un ID d'autorisation. Cet ID fait référence à l'évaluation de l'autorisation pour cette demande refusée. Pour récupérer les détails de l'évaluation et identifier la cause du refus, envoyez une demande à l'`GetRequestAuthorizationDetails`API et fournissez l'ID d'autorisation, ou ouvrez le lien présent dans le message d'erreur. Cela vous permet d'identifier la cause d'une demande refusée en un seul appel, plutôt que de résoudre les refus un par un.

> **Note**
>
> AWS ne garantit pas un identifiant d'autorisation pour chaque demande refusée. Voir Lorsque vous ne recevez pas d'identifiant d'autorisation dans ce document.

L'exemple suivant montre un message d'erreur d'accès refusé qui inclut un ARN de politique, un lien vers la console de AWS gestion et un ID d'autorisation :

```
User: arn:aws:iam::123456789012:user/janedoe is not authorized to perform:
 iam:CreateInstanceProfile
on resource: arn:aws:iam::123456789012:instance-profile/my-instance-profile
with an explicit deny in an identity-based policy: arn:aws:iam::123456789012:policy/
ExampleDenyPolicy.
Go to https://us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/
authorization-details/EXAMPLE1a2b3c4d5e6f7g8h9i0j for complete details,
or call the GetRequestAuthorizationDetails API with the following authorization id:
 EXAMPLE1a2b3c4d5e6f7g8h9i0j
```

Concepts et définitions

- Détails de l'autorisation. Les détails de l'autorisation incluent le contexte de la demande, les évaluations effectuées et les politiques qui ont été évaluées.
- ID d'autorisation. Identifiant unique inclus dans une réponse d'erreur d'accès refusé prise en charge. Il fait référence à l'évaluation de l'autorisation pour cette demande refusée et correspond à la valeur que vous transmettez au lien AWS de la console de gestion ou à l'`GetRequestAuthorizationDetails`API.
- Contexte de la demande. Les valeurs des clés de contexte conditionnelles évaluées pour la demande. Cela inclut les clés de contexte de condition globales et les clés de contexte de condition spécifiques au service.
- Évaluation. Pour chaque action/resource paire de la demande, les clés de contexte des conditions appliquées et le résultat évalué :`ALLOW`, `EXPLICIT_DENY` (une politique a explicitement refusé l'action), ou `IMPLICIT_DENY` (aucune politique n'a autorisé l'action), et une référence à chaque stratégie et instruction correspondant à la action/resource paire. Chaque référence est un ARN de politique (pour les politiques gérées) ou un URI (pour les politiques en ligne et autres politiques non gérées) qui est réutilisé dans la réponse.
- Politiques évaluées lors de l'autorisation. Les politiques AWS prises en compte pour la demande, y compris les politiques qui ne correspondaient pas à la action/resource paire. Chaque entrée indique le type de politique (par exemple, basée sur l'identité, basée sur les ressources, SCP, RCP, limite d'autorisations ou politique de session) et le même ARN (pour les politiques gérées) ou URI que celui utilisé dans les évaluations (pour les politiques en ligne et autres politiques non gérées). Utilisez l'ARN ou l'URI pour faire correspondre une politique à l'évaluation qui l'a référencée. Le champ « Attaché à » indique l'ARN principal du principal auquel la politique évaluée est attachée.

Utilisez ces résultats conjointement pour identifier quelle politique a refusé l'accès et pourquoi. L'évaluation reflète les politiques telles qu'elles étaient au moment du refus. Les politiques ont peut-être changé depuis lors.

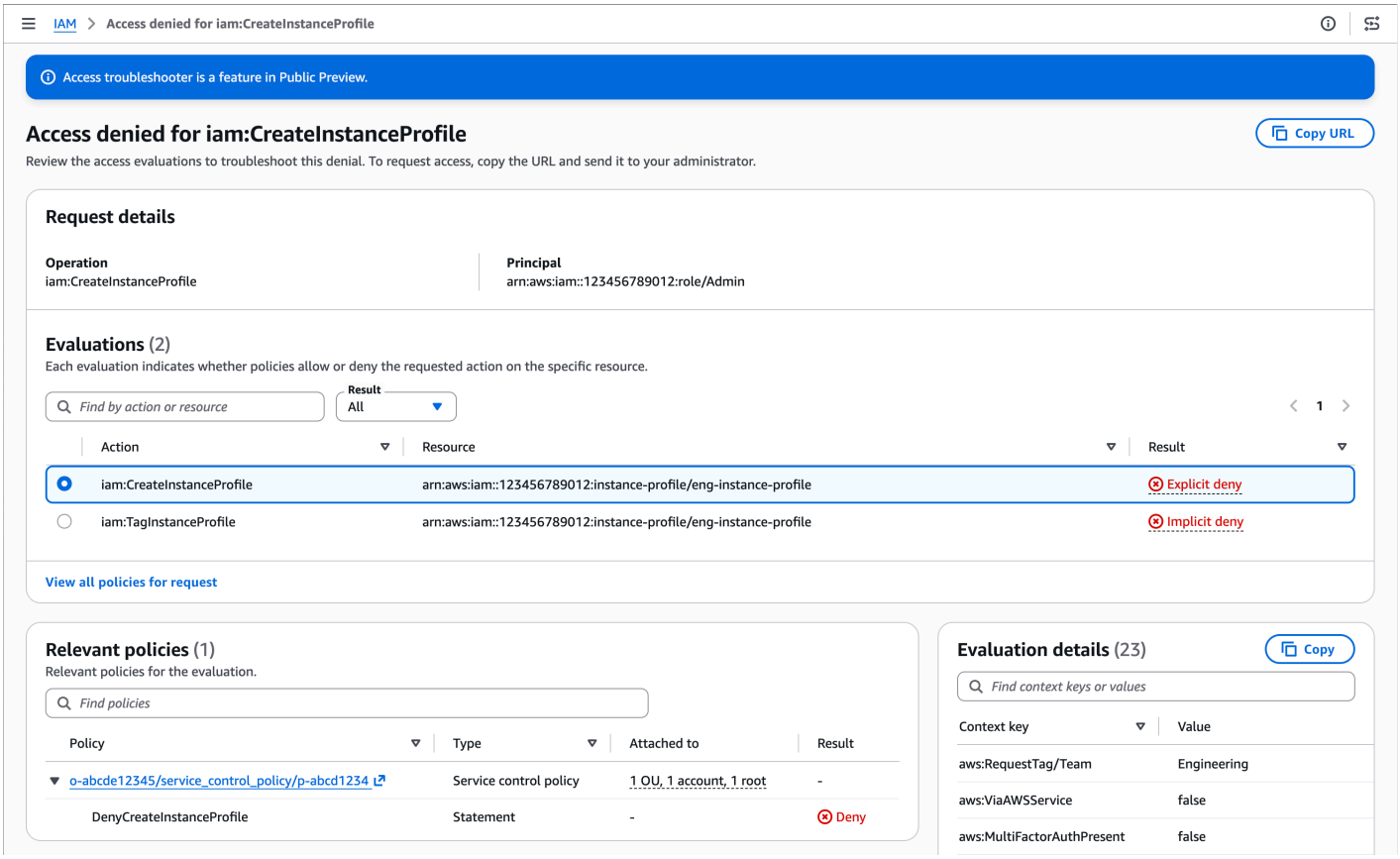
AWS inclut automatiquement un identifiant d'autorisation dans les réponses d'erreur d'accès refusé pour les API prises en charge, mais les identifiants d'autorisation ne sont pas garantis. Dans certains cas, AWS n'enregistre pas les détails d'autorisation en cas de refus et la réponse d'erreur n'inclut pas d'ID d'autorisation. Lorsque des informations sont enregistrées, elles peuvent ne pas être disponibles immédiatement, car elles sont AWS enregistrées de manière asynchrone. Si un appel `GetRequestAuthorizationDetails` renvoie une réponse introuvable, patientez quelques

instants et réessayez. AWS conserve les informations d'autorisation pendant 24 heures après le refus.

Résoudre les problèmes à l'aide du AWS Console de gestion

Cliquez sur le lien figurant dans le message d'erreur d'accès refusé pour ouvrir les détails de l'autorisation dans la console de résolution des problèmes d'accès.

- Ce qu'il faut rechercher. La console Access Troubleshooter affiche l'évaluation de chaque action/resource paire de la demande, le résultat de chacune d'elles (`AllowExplicit deny`, ou`Implicit deny`) et les politiques prises en compte. Trouvez la paire qui a été refusée et la politique à l'origine du refus. Consultez la capture d'écran suivante à titre d'exemple :



- Déterminez si le refus était intentionnel. Passez en revue l'évaluation avant de modifier une politique. Un refus peut être correct, et la cause en est parfois le contexte de la demande plutôt que la politique. Par exemple, il se peut qu'une requête ne comporte pas une balise de session attendue ou qu'elle provienne d'un chemin réseau inattendu.
- Comment résoudre le refus. Localisez la politique correspondante et mettez-la à jour pour résoudre le problème d'accès. En cas de refus explicite, supprimez ou limitez la `Deny` déclaration

relative à l'action. Pour un refus implicite, ajoutez une `Allow` déclaration pour l'action. Pour de plus amples informations, veuillez consulter Modification de politiques IAM.

Qui peut accéder aux informations d'autorisation. Pour une demande au sein d'un seul compte ou d'une seule organisation, toute personne à qui l'accès est autorisé `iam:GetRequestAuthorizationDetails` peut accéder aux détails de la demande. Pour une demande qui concerne plusieurs organisations, un utilisateur d'une organisation donnée ne peut voir que les informations relatives à sa propre organisation.

Résoudre les problèmes en utilisant AWS API ou AWS CLI

Pour résoudre un refus à l'aide de l' AWS API ou AWS CLI :

1. Localisez l'ID d'autorisation dans la réponse d'erreur d'accès refusé.
2. Appelez `GetRequestAuthorizationDetails` avec l'ID d'autorisation, à l' Région AWS endroit même où le refus s'est produit. L'exemple suivant montre la AWS CLI commande :

```
aws iam-toolbox get-request-authorization-details \
    --region us-east-1 \
    --authorization-id EXAMPLE1a2b3c4d5e6f7g8h9i0j
```

3. Passez en revue les résultats de l'évaluation pour identifier les politiques qui ont refusé l'accès et la raison de ce refus.
4. Mettez à jour les politiques identifiées pour résoudre le problème d'accès. Pour de plus amples informations, veuillez consulter Modification de politiques IAM.

Lorsque vous ne recevez pas d'identifiant d'autorisation

AWS ne renvoie pas d'identifiant d'autorisation dans les cas suivants :

- L'autorisation est trop importante pour être enregistrée. Lorsqu'une demande implique un grand nombre de politiques, vous AWS pouvez décider de ne pas l'enregistrer.
- Le service n'est pas encore pris en charge. Consultez Services pris en charge.
- L'API n'est pas prise en charge. Certaines API d'un service pris en charge ne renvoient pas d'identifiant d'autorisation.

- Le type de demande n'est pas pris en charge. Cela inclut les demandes faites par un service en votre nom et les demandes anonymes.

Si vous ne recevez pas d'identifiant d'autorisation, découvrez Résolution des problèmes liés aux messages d'erreur d'accès rejeté d'autres moyens de diagnostiquer le refus.

Considérations supplémentaires

Autorisations requises. Pour appeler`GetRequestAuthorizationDetails`, vous devez disposer des autorisations nécessaires pour effectuer l'`iam:GetRequestAuthorizationDetails`action.

API régionale. `GetRequestAuthorizationDetails`est une API régionale. Appelez-le de la même manière Région AWS que l'erreur d'accès refusé et l'ID d'autorisation ont été générés.

Cross-organization demandes. Lorsqu'une demande refusée concerne plus d'une organisation, `GetRequestAuthorizationDetails` renvoie uniquement les informations qui appartiennent à votre propre organisation. Cela vaut dans les deux sens : si vous faites partie de l'organisation de l'appelant refusé, vous ne voyez que les informations relatives à cette organisation. Si vous faites partie de l'organisation de la ressource refusée, seuls les détails relatifs à cette organisation s'affichent. Les informations appartenant à une autre organisation, telles que les politiques de cette organisation ou les valeurs clés contextuelles de cette organisation, ne sont pas renvoyées.

Si vous recevez une erreur d'accès refusé provenant d'une API, d'un service ou d'une région qui n'inclut pas encore d'identifiant d'autorisation, consultezRésolution des problèmes liés aux messages d'erreur d'accès rejeté.

Services pris en charge

La prise en charge de `GetRequestAuthorizationDetails` est déployée progressivement sur AWS les API et Région AWS. Le (s) service (s) suivant (s) prend en charge `GetRequestAuthorizationDetails` :

- La plupart des API dans IAM

Résolution de problèmes relatifs à l'utilisateur racine

Utilisez les informations fournies pour vous aider à résoudre les problèmes liés à l'utilisateur root d'un Compte AWS.

> **Note**
>
> Comptes AWS géré à l'aide d'un accès root centralisé AWS Organizations peut être activé pour les comptes des membres. Ces comptes membres ne disposent pas d'informations d'identification de l'utilisateur racine, ne peuvent pas se connecter en tant qu'utilisateur racine et ne peuvent pas récupérer le mot de passe de l'utilisateur racine. Contactez votre administrateur si vous devez effectuer une tâche qui nécessite les informations d'identification de l'utilisateur racine.

Je ne peux pas effectuer les tâches que je pense pouvoir effectuer lorsque je suis connecté en tant qu'utilisateur root du compte

Votre compte doit être membre de votre organisation dans AWS Organizations. L'administrateur de votre organisation dispose peut-être d'une politique de contrôle des services (SCP) limitant les autorisations de votre compte. SCPs ont un impact sur tous les utilisateurs, y compris l'utilisateur root. Pour plus d'informations, consultez Politiques de contrôle de service dans le Guide de l'utilisateur AWS Organizations .

J'ai oublié le mot de passe utilisateur root de mon Compte AWS

Si vous êtes un utilisateur root et que vous avez perdu ou oublié le mot de passe de votre Compte AWS, vous pouvez réinitialiser votre mot de passe. Vous devez connaître l'adresse e-mail utilisée pour créer le Compte AWS, et vous devez avoir accès au compte e-mail. Pour de plus amples informations, veuillez consulter Réinitialisation d'un mot de passe d'utilisateur racine perdu ou oublié.

Je n'ai pas accès à l'e-mail de mon Compte AWS

Lorsque vous créez un Compte AWS, vous fournissez une adresse e-mail et un mot de passe. Ce sont les informations d'identification pour le Utilisateur racine d'un compte AWS. Si vous n'êtes pas sûr de l'adresse e-mail associée à votre Compte AWS, recherchez les messages envoyés depuis `@signin.aws` ou `@verify.signin.aws` vers une adresse e-mail de votre organisation qui aurait pu être utilisée pour ouvrir le Compte AWS.

Si vous connaissez l'adresse e-mail, mais que vous n'avez plus accès à l'e-mail, essayez de récupérer l'accès à l'e-mail. Utilisez l'une des options suivantes pour retrouver l'accès à votre e-mail :

- Si vous possédez le domaine pour l'adresse e-mail, vous pouvez restaurer une adresse e-mail supprimée. Sinon, vous pouvez configurer une adresse fourre-tout pour votre compte

de messagerie. Un fourre-tout collecte tous les messages envoyés à des adresses e-mail qui n'existent plus dans le serveur de messagerie et les redirige vers une autre adresse e-mail.

- Si l'adresse e-mail sur le compte fait partie de votre système de messagerie d'entreprise, nous vous recommandons de contacter vos administrateurs de système informatique. Ils peuvent vous aider à récupérer l'accès à l'e-mail.

Si vous ne parvenez toujours pas à vous connecter à votre compte Compte AWS, vous pouvez trouver d'autres options d'assistance sur Contactez-nous.

Dépannage de politiques IAM

Une politique est une entité AWS qui, lorsqu'elle est associée à une identité ou à une ressource, définit ses autorisations. AWS évalue ces politiques lorsqu'un mandant, tel qu'un utilisateur, fait une demande. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée.

Les politiques sont stockées AWS sous forme de documents JSON qui sont joints aux principaux en tant que politiques basées sur l'identité ou aux ressources en tant que politiques basées sur les ressources. Identity-based les politiques incluent les politiques AWS gérées, les politiques gérées par le client et les politiques en ligne.

Vous pouvez créer et modifier des politiques gérées par le client à l' Console de gestion AWS aide des options de l'éditeur Visual et JSON. Lorsque vous consultez une politique dans le Console de gestion AWS, vous pouvez voir un résumé des autorisations accordées par cette politique. Vous pouvez utiliser l'éditeur visuel et des résumés de politique pour vous aider à diagnostiquer et à corriger les erreurs courantes lors de la gestion des politiques IAM.

N'oubliez pas que toutes les politiques IAM sont stockées selon une syntaxe qui commence par les règles de notation d'JavaScript objet (JSON). Vous n'avez pas besoin de comprendre cette syntaxe pour créer ou gérer vos politiques. Vous pouvez créer et modifier une politique à l'aide de l'éditeur visuel dans l' Console de gestion AWS. Pour en savoir plus sur la syntaxe JSON dans les politiques IAM, veuillez consulter Syntaxe du langage de politique JSON IAM.

Rubriques de résolution des erreurs de politique IAM

- Résolution des problèmes à l'aide de l'éditeur visuel
  - Restructuration de politique
  - Choix d'un ARN de ressource dans l'éditeur visuel

- Refus des autorisations dans l'éditeur visuel
- Spécification de plusieurs services dans l'éditeur visuel
- Réduction de la taille de votre politique dans l'éditeur visuel
- Résolution des problèmes de services, d'actions ou de types de ressources non reconnus dans l'éditeur visuel
- Résolution des problèmes avec les récapitulatifs de politique
  - Récapitulatif de politique manquant
  - Le récapitulatif de politique inclut des services, des actions ou des types de ressource non reconnus
  - Le service ne prend pas en charge les récapitulatifs de politique IAM
  - Ma politique n'accorde pas les autorisations escomptées
- Résolution des problèmes de gestion des politiques
  - Attacher ou détacher une politique dans un compte IAM
  - Modification des politiques pour vos identités IAM en fonction de leur activité
- Résolution des problèmes de documents de politique JSON
  - Valider vos politiques
  - Je n'ai pas d'autorisations pour la validation de politique dans l'éditeur JSON
  - Plusieurs objets de politique JSON
  - Plusieurs éléments de instruction JSON
  - Plusieurs éléments Effect, Action ou Resource dans un élément d'instruction JSON
  - Élément de version JSON manquant

Résolution des problèmes à l'aide de l'éditeur visuel

Lorsque vous créez ou modifiez une politique gérée par le client, vous pouvez utiliser les informations de l'éditeur visuel pour vous aider à résoudre les erreurs dans votre politique. Pour voir un exemple d'utilisation de l'éditeur visuel pour la création d'un politique, veuillez consulter the section called "Contrôle de l'accès aux identités".

Restructuration de politique

Lorsque vous créez une politique, vous la AWS validez, la traitez et la transformez avant de la stocker. Lorsque la politique est récupérée, elle AWS redevient un format lisible par l'homme sans

modifier les autorisations. Cela peut entraîner des différences quant à ce que vous voyez dans l’éditeur visuel de politique ou sur l’onglet JSON.

- Les blocs d’autorisation de l’éditeur visuel peuvent être ajoutés, supprimés ou réorganisés, et le contenu d’un bloc peut être optimisé.
- Sous l'onglet JSON, les espaces blancs sans importance peuvent être supprimés et les éléments avec des mappes JSON peuvent être réorganisés. En outre, Compte AWS les identifiants figurant dans les éléments principaux peuvent être remplacés par le nom de ressource Amazon (ARN) du Utilisateur racine d'un compte AWS.

En raison de ces changements possibles, vous ne devez pas comparer les documents de politique JSON sous forme de chaînes.

Lorsque vous créez une politique gérée par le client dans le Console de gestion AWS, vous pouvez choisir de travailler entièrement dans l'éditeur JSON. Si vous ne modifiez jamais la politique dans l’éditeur Visuel et si vous choisissez Suivant dans l’éditeur JSON, la politique a moins de chances d’être restructurée. Lorsque vous utilisez l’éditeur Visuel, IAM peut restructurer la politique afin d’optimiser son apparence. Cette restructuration existe uniquement dans votre session d'édition et n'est pas enregistrée automatiquement.

Si votre politique est restructurée dans votre session d'édition, IAM détermine si la restructuration doit être enregistrée en fonction des situations suivantes :

Utilisation de cette option de l'éditeur	Si vous modifiez votre politique	Et puis choisisse z Suivant dans cet onglet	Lorsque vous sélectionnez Enregistr er les modifications
Visuel	Modifié	Visuel	La stratégie est restructurée
Visuel	Modifié	JSON	La stratégie est restructurée
Visuel	Non modifié	Visuel	La stratégie est restructurée
JSON	Modifié	Visuel	La stratégie est restructurée

Utilisation de cette option de l'éditeur	Si vous modifiez votre politique	Et puis choisisse z Suivant dans cet onglet	Lorsque vous sélectionnez Enregistr er les modifications
JSON	Modifié	JSON	La structure de la stratégie n'est pas modifiée
JSON	Non modifié	JSON	La structure de la stratégie n'est pas modifiée

IAM peut restructurer les politiques complexes ou celles qui ont des blocs d'autorisations ou des instructions qui permettent plusieurs services, types de ressources ou clés de condition.

Choix d'un ARN de ressource dans l'éditeur visuel

Lorsque vous créez ou modifiez une politique à l'aide de l'éditeur visuel, vous devez d'abord choisir un service, puis les actions de ce dernier. Si le service et les actions que vous avez sélectionnés prennent en charge la sélection de ressources spécifiques, l'éditeur visuel répertorie les types de ressources pris en charge. Vous pouvez ensuite choisir Add ARN (Ajouter un ARN) pour fournir des détails sur votre ressource. Vous pouvez choisir l'une des options suivantes afin d'ajouter un ARN pour un type de ressource.

- Utiliser le générateur d’ARN : selon le type de ressource, différents champs peuvent s’afficher pour créer votre ARN. Vous pouvez également choisir Any (Tout) pour fournir des autorisations pour n'importe quelle valeur du paramètre spécifié. Par exemple, si vous avez sélectionné le groupe de niveau d'accès Read (Lecture) Amazon EC2, les actions de votre politique prennent en charge le type de ressource `instance`. Indiquez la région, le compte et InstanceId les valeurs de votre ressource. La politique accorde des autorisations à toutes les instances de votre compte si vous fournissez l’ID de votre compte et choisissez Tout pour la région et de l’ID de l’instance.
- Saisir ou coller l'ARN : vous pouvez spécifier les ressources par leur Amazon Resource Name (ARN). Vous pouvez inclure un caractère générique (*****) dans n'importe quel champ de l'ARN (entre chaque paire de signes deux-points). Pour de plus amples informations, veuillez consulter Éléments de la politique JSON IAM : Ressource.

Refus des autorisations dans l'éditeur visuel

Par défaut, la politique que vous créez à l'aide de l'éditeur visuel autorise les actions que vous sélectionnez. Pour refuser les actions choisies, sélectionnez Switch to deny permissions (Basculer vers le refus des autorisations). Dans la mesure où les demandes sont refusées par défaut, nous vous recommandons d'autoriser un utilisateur à accéder uniquement aux actions et aux ressources dont il a besoin. Vous devez créer une instruction de refus des autorisations uniquement si vous souhaitez remplacer une autorisation qui est également autorisée par une autre instruction ou politique. Nous vous recommandons de limiter le nombre de refus d'autorisation au minimum, car ils peuvent rendre la résolution des problèmes d'autorisation plus complexe. Pour plus d'informations sur la logique d'évaluation de politique IAM, veuillez consulter Logique d'évaluation de politiques.

> **Note**
>
> Par défaut, seul le Utilisateur racine d'un compte AWS a accès à toutes les ressources de ce compte. Par conséquent, si vous n'êtes pas connecté en tant qu'utilisateur racine, vous devez disposer des autorisations accordées par une politique.

Spécification de plusieurs services dans l'éditeur visuel

Lorsque vous utilisez l'éditeur visuel pour construire une politique, vous pouvez sélectionner un seul service à la fois. Il s'agit d'une bonne pratique car l'éditeur visuel vous permet ensuite de choisir parmi les actions associées à ce service. Vous pouvez ensuite choisir parmi les ressources prises en charge par ce service et les actions sélectionnées. Cela facilite la création et la résolution des problèmes de votre politique.

Vous pouvez également utiliser un caractère générique (*) afin de spécifier manuellement plusieurs services. Par exemple, saisissez **`Code*`** pour fournir des autorisations pour tous les services commençant par `Code`, tels que `CodeBuild` et `CodeCommit`. Toutefois, vous devez ensuite taper les ARN des actions et ressources pour finaliser votre politique. De plus, lorsque vous enregistrez votre politique, celle-ci peut être restructurée de façon à inclure chaque service dans un bloc d'autorisation distinct.

Afin d'utiliser la syntaxe JSON (comme les caractères génériques) pour les services, créez, modifiez et enregistrez votre politique à l'aide de l'option de l'éditeur JSON.

Réduction de la taille de votre politique dans l'éditeur visuel

Lorsque vous utilisez l'éditeur visuel pour créer une politique, IAM crée un document JSON pour stocker votre politique. Vous pouvez afficher ce document basculant vers l'option de l'éditeur JSON. Si ce document JSON dépasse la limite de taille d'une politique, l'éditeur visuel affiche un message d'erreur. Vous ne pourrez pas consulter ni enregistrer la politique. Pour afficher la limitation IAM relative à la taille d'une politique gérée, veuillez consulter Limites des caractères d'IAM et de STS.

Pour réduire la taille de votre politique dans l'éditeur visuel, modifiez votre politique ou déplacez les blocs d'autorisation dans une autre politique. Le message d'erreur inclut le nombre de caractères que votre document de politique contient. Vous pouvez utiliser ces informations pour vous aider à réduire la taille de votre politique.

Résolution des problèmes de services, d'actions ou de types de ressources non reconnus dans l'éditeur visuel

Il se peut que l'éditeur visuel affiche un avertissement indiquant que votre politique inclut un type de service, d'action ou de ressource non reconnu.

> **Note**
>
> IAM vérifie les noms de services, les actions et les types de ressource pour les services qui prennent en charge les récapitulatifs de politique. Toutefois, le récapitulatif de votre politique peut contenir une valeur de ressource ou une condition qui n'existe pas. Testez toujours vos stratégies avec le simulateur de stratégie.

Si votre stratégie inclut des services, des actions ou des types de ressource non reconnus, une des erreurs suivantes s'est produite :

- Service d'aperçu : les services en mode aperçu ne prennent pas en charge l'éditeur visuel. Si vous participez à l'aperçu, vous devez taper manuellement les ARN d'actions et de ressources pour finaliser votre politique. Vous pouvez ignorer les avertissements et continuer. Vous pouvez également choisir l'option éditeur JSON afin de taper ou de coller un document de politique JSON.
- Service personnalisé : les services personnalisés ne prennent pas en charge l'éditeur visuel. Si vous utilisez un service personnalisé, vous devez taper manuellement les ARN d'actions et de ressources pour finaliser votre politique. Vous pouvez ignorer les avertissements et continuer. Vous pouvez également choisir l'option éditeur JSON afin de taper ou de coller un document de politique JSON.

- Le service ne prend pas en charge l’éditeur visuel : si votre politique inclut un service disponible au public (GA) qui ne prend pas en charge l’éditeur visuel, vous devez taper manuellement les ARN d’actions et de ressources pour finaliser votre politique. Vous pouvez ignorer les avertissements et continuer. Vous pouvez également choisir l'option éditeur JSON afin de taper ou de coller un document de politique JSON.

  Les services disponibles pour tous sont des services mis à disposition du public qui ne sont pas des services d'aperçu ou personnalisés. Si un service non reconnu est disponible pour tous et que son nom est orthographié correctement, il ne prend pas en charge l'éditeur visuel. Pour savoir comment demander la prise en charge de l'éditeur visuel ou du résumé de stratégie pour un service disponible pour tous (GA), consultez Le service ne prend pas en charge les récapitulatifs de politique IAM.
- L’action ne prend pas en charge l’éditeur visuel : si votre politique inclut un service pris en charge avec une action non prise en charge, vous devez taper manuellement les ARN d’actions et de ressources pour finaliser votre politique. Vous pouvez ignorer les avertissements et continuer. Vous pouvez également choisir l'option éditeur JSON afin de taper ou de coller un document de politique JSON.

  Si votre politique contient un service pris en charge avec une action non prise en charge, le service ne prend pas en charge intégralement l'éditeur visuel. Pour savoir comment demander la prise en charge de l'éditeur visuel ou du résumé de stratégie pour un service disponible pour tous (GA), consultez Le service ne prend pas en charge les récapitulatifs de politique IAM.
- Le type de ressource ne prend pas en charge l'éditeur visuel : si votre politique inclut une action prise en charge avec un type de ressource non pris en charge, vous pouvez ignorer le message d'avertissement et poursuivre. Toutefois, IAM ne peut pas confirmer que vous avez inclus des ressources pour l'ensemble de vos actions sélectionnées et vous pouvez consulter des avertissements supplémentaires.
- Faute de frappe : lorsque vous saisissez manuellement un service, une action ou une ressource dans l'éditeur visuel, vous pouvez parfois créer une politique qui inclut une faute de frappe. Nous vous recommandons d’utiliser l’éditeur visuel en sélectionnant dans la liste des services et des actions. Complétez ensuite la section des ressources conformément aux invites. Si un service ne prend pas totalement en charge l’éditeur visuel, vous devrez peut-être taper manuellement certaines parties de votre politique.

  Si vous êtes sûr que votre politique ne contient aucune des erreurs ci-dessus, il se peut que votre politique contienne une faute de frappe. Vérifiez les problèmes suivants :

- Noms de type de service, d'action et de ressource mal orthographiés, tels que `s2` au lieu `s3` ou `ListMyBuckets` au lieu de `ListAllMyBuckets`
- Texte inutile dans les ARN, tel que `arn:aws:s3: : :*`
- Deux points manquants dans des actions, telles que `iam.CreateUser`

Vous pouvez évaluer une politique susceptible de contenir des fautes de frappe en choisissant Suivant afin de vérifier le récapitulatif. Ensuite, confirmez que la politique contient les autorisations prévues.

Résolution des problèmes avec les récapitulatifs de politique

Vous pouvez diagnostiquer et résoudre les problèmes liés aux résumés de politique.

Récapitulatif de politique manquant

La console IAM comprend des tables de récapitulatif de la politique qui présentent le niveau d'accès, les ressources et les conditions autorisées ou rejetées pour chaque service dans une politique. Les politiques sont résumées dans trois tables : récapitulatif de la politique, récapitulatif du service et récapitulatif de l'action.

La table récapitulative de la politique comprend une liste des services et des résumés des autorisations définies par la politique choisie. Le résumé de la politique est disponible sur la page Détails de la politique pour toute stratégie associée, ou sur la page Politiques pour les politiques gérées. Si vous AWS ne parvenez pas à afficher le résumé d'une politique, la console affiche le document de stratégie JSON et l'erreur suivante :

Aucun récapitulatif pour cette politique ne peut être généré. Vous pouvez toujours afficher ou modifier le document de politique JSON.

Si votre politique ne comprend pas de récapitulatif, une des erreurs suivantes s'est produite :

- Élément de politique non pris en charge : IAM ne prend pas en charge la génération de récapitulatifs de politique pour les politiques comprenant l'un des éléments de politique suivants :
  - `Principal`
  - `NotPrincipal`
  - `NotResource`
- Aucune autorisation de politique : si une politique ne fournit pas d'autorisations effectives, le récapitulatif de politique ne peut pas être généré. Par exemple, si une politique contient une

instruction unique avec l'élément `"NotAction": "*"`, cela signifie qu'elle accorde l'accès à toutes les actions, sauf « toutes les actions »(*). Cela signifie qu'elle accorde l'accès `Deny` ou `Allow` à rien.

> **Note**
>
> Soyez prudent lorsque vous utilisez ces éléments de politique tels que `NotPrincipal`, `NotAction` et `NotResource`. Pour en savoir plus sur l'utilisation des éléments de la politique, veuillez consulter Référence de l'élément de politique JSON IAM.

Vous pouvez créer une politique qui ne fournit pas d'autorisations effectives si vous fournissez des services et des ressources non appariés. Cela peut se produire lorsque vous précisez des actions dans un service et des ressources provenant d'un autre service. Dans ce cas, le récapitulatif de la politique n'apparaît pas. La seule indication qu'un problème est survenu est que la colonne de ressource du récapitulatif contient une ressource provenant d'un service différent. Si cette colonne contient une ressource non appariée, recherchez d'éventuelles erreurs dans votre politique. Afin de mieux comprendre la politique, testez-la avec le simulateur de politique.

Le récapitulatif de politique inclut des services, des actions ou des types de ressource non reconnus

Dans la console IAM, si un récapitulatif de politique inclut un symbole d'avertissement ( ), la politique peut alors inclure un service, une action ou un type de ressource non reconnu. Pour en savoir plus sur les avertissements dans un récapitulatif de politique, veuillez consulter Récapitulatif de la politique (liste des services).

> **Note**
>
> IAM vérifie les noms de services, les actions et les types de ressource pour les services qui prennent en charge les récapitulatifs de politique. Toutefois, le récapitulatif de votre politique peut contenir une valeur de ressource ou une condition qui n'existe pas. Testez toujours vos stratégies avec le simulateur de stratégie.

Si votre stratégie inclut des services, des actions ou des types de ressource non reconnus, une des erreurs suivantes s'est produite :

- Service d'aperçu : les services en mode aperçu ne prennent pas en charge les récapitulatifs de politique.
- Service personnalisé : les services personnalisés ne prennent pas en charge les récapitulatifs de politique.
- Le service ne prend pas en charge les récapitulatifs : si votre politique contient un service disponible pour tous (GA) qui ne prend pas en charge les récapitulatifs de politique, le service figure dans la section Unrecognized services (Services non reconnus) de la table récapitulative de la politique. Les services disponibles pour tous sont des services mis à disposition du public qui ne sont pas des services d'aperçu ou personnalisés. Si un service non reconnu est disponible pour tous est que son nom est orthographié correctement, il ne prend pas en charge les récapitulatifs de politique IAM. Pour savoir comment demander la prise en charge du récapitulatif de politique d'un service disponible pour tous (GA), veuillez consulter Le service ne prend pas en charge les récapitulatifs de politique IAM.
- L'action ne prend pas en charge les récapitulatifs : si votre politique contient un service pris en charge avec une action non prise en charge, l'action figure alors dans la section Unrecognized actions (Actions non reconnues) de la table récapitulative du service. Pour en savoir plus sur les avertissements dans un récapitulatif de service, consultez Récapitulatif du service (liste des actions).
- Le type de ressource ne prend pas en charge les récapitulatifs : si votre politique inclut une action prise en charge avec un type de ressource non pris en charge, la ressource figure dans la section Unrecognized resource types (Types de ressource non reconnus) de la table récapitulative du service. Pour en savoir plus sur les avertissements dans un récapitulatif de service, consultez Récapitulatif du service (liste des actions).
- Typo : AWS vérifie que le JSON est syntaxiquement correct et que la politique n'inclut pas de fautes de frappe ou d'autres erreurs dans le cadre de la validation de la politique. ???

> **Note**
>
> Selon les bonnes pratiques, nous vous recommandons d'utiliser IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sûres et fonctionnelles. Nous vous recommandons d'ouvrir vos politiques existantes, d'examiner et de résoudre toutes les recommandations de validation des politiques.

Le service ne prend pas en charge les récapitulatifs de politique IAM

Il est possible que les récapitulatifs de politique IAM ou l'éditeur visuel ne prennent pas en charge un service ou une action disponible pour tous. Les services disponibles pour tous sont des services mis à disposition du public qui ne sont pas des services d'aperçu ou personnalisés. Si un service non reconnu est disponible pour tous et que son nom est orthographié correctement, il ne prend pas en charge ces fonctionnalités. Si votre politique contient un service pris en charge avec une action non prise en charge, le service ne prend pas en charge intégralement les récapitulatifs de politique IAM.

Pour demander à ce qu'un service ajoute la prise en charge du résumé de politique IAM ou de l'éditeur visuel

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Recherchez la stratégie incluant le service non pris en charge :
   - Si la stratégie est une stratégie gérée, choisissez Stratégies dans le panneau de navigation. Dans la liste des stratégies, choisissez le nom de la stratégie à afficher.
   - Si la stratégie est une stratégie en ligne attachée à l'utilisateur, choisissez Utilisateurs dans le panneau de navigation. Dans la liste des utilisateurs, choisissez le nom de l'utilisateur dont vous souhaitez afficher la politique. Dans le tableau des politiques dédiées à l'utilisateur, développez l'en-tête du récapitulatif de politique à afficher.
3. Dans la partie gauche du Console de gestion AWS pied de page, choisissez Feedback. Dans la case Commentaires pour IAM, tapez **`I request that the <ServiceName> service add support for IAM policy summaries and the visual editor`**. Si vous avez besoin de plusieurs services pour prendre en charge les récapitulatifs, saisissez **`I request that the <ServiceName1>, <ServiceName2>, and <ServiceName3> services add support for IAM policy summaries and the visual editor`**.

Pour demander que la prise en charge du récapitulatif de politique IAM d'une action manquante soit ajoutée à un service

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Recherchez la stratégie incluant le service non pris en charge :

- Si la stratégie est une stratégie gérée, choisissez Stratégies dans le panneau de navigation. Dans la liste des stratégies, choisissez le nom de la stratégie à afficher.
- Si la stratégie est une stratégie en ligne attachée à l'utilisateur, choisissez Utilisateurs dans le panneau de navigation. Dans la liste des utilisateurs, choisissez le nom de l'utilisateur dont vous souhaitez afficher la politique. Dans le tableau des politiques dédiées à l'utilisateur, sélectionnez le nom de la politique à afficher pour développer le récapitulatif de politique.

3. Dans le récapitulatif de la politique, sélectionnez le nom du service incluant une action non prise en charge.
4. Dans la partie gauche du Console de gestion AWS pied de page, choisissez Feedback. Dans la case Commentaires pour IAM, tapez **`I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName> action`**. Si vous souhaitez signaler plusieurs actions non prises en charge, saisissez **`I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName1>, <ActionName2>, and <ActionName3> actions`**.

Pour demander qu'un service différent contienne des actions manquantes, répétez les trois dernières étapes.

Ma politique n'accorde pas les autorisations escomptées

Pour attribuer des autorisations à un utilisateur, un groupe, un rôle ou une ressource, vous devez créer une politique, autrement dit un document qui définit les autorisations. Le document de politique inclut les éléments suivants :

- Effet : si la politique autorise ou refuse l'accès
- Action : la liste des actions qui sont autorisées ou rejetées par la politique
- Ressource : la liste des ressources sur lesquelles les actions peuvent se produire
- Condition (Facultatif) : les circonstances dans lesquelles la politique accorde l'autorisation

Pour en savoir plus sur ces éléments et d'autres éléments d'une politique, veuillez consulter Référence de l'élément de politique JSON IAM.

Pour accorder l'accès, votre politique doit définir une action avec une ressource prise en charge. Si votre politique inclut également une condition, cette condition doit inclure une clé de condition ou doit

s'appliquer à l'action. Pour connaître les ressources qu'une action prend en charge, veuillez consulter la documentation AWS correspondant à votre service. Pour savoir quelles conditions sont prises en charge par une action, voir Actions, ressources et clés de condition pour les AWS services.

Vérifiez que votre politique définit une action, une ressource ou une condition qui n'accorde pas d'autorisations. Consultez le résumé de votre politique à l'aide de la console IAM à https://console.aws.amazon.com/iam/ l'adresse. Vous pouvez utiliser les récapitulatifs de politiques afin d'identifier et de corriger les problèmes dans votre politique.

Plusieurs raisons peuvent expliquer l'incapacité d'un élément à accorder des autorisations bien que celles-ci soient définies dans la politique IAM :

- Une action est définie sans ressource applicable
- Une ressource est définie sans action applicable
- Une condition est définie sans action applicable

Pour voir des exemples de récapitulatifs de politiques incluant des avertissements, veuillez consulter the section called "Récapitulatif de la politique (liste des services)".

Une action est définie sans ressource applicable

La politique ci-dessous définit toutes les actions `ec2:Describe*` et définit une ressource spécifique. Aucune des `ec2:Describe` actions n'est accordée car aucune de ces actions ne prend en charge les autorisations au niveau des ressources. Resource-level les autorisations signifient que l'action prend en charge les ressources utilisant des ARN dans l'`Resource`élément de la politique. Si une action ne prend pas en charge les autorisations de niveau ressource, cette instruction de la politique doit utiliser le caractère générique (`*`) dans l'élément `Resource`. Pour savoir quels services prennent en charge les autorisations au niveau des ressources, veuillez consulter AWS services qui fonctionnent avec IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
```

```
            "Resource": "arn:aws:ec2:us-east-2:111122223333:instance/*"
        }
    ]
}
```

Cette politique ne fournit pas d'autorisations, et le récapitulatif de politique inclut l'erreur suivante :

`This policy does not grant any permissions. To grant access, policies must have an action that has an applicable resource or condition.`

Pour corriger cette politique, vous devez utiliser le caractère générique * dans l'élément `Resource`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
    }]
}
```

Une ressource est définie sans action applicable

La politique ci-dessous définit une ressource de compartiment Amazon S3, mais n'inclut pas d'action S3 pouvant être effectuée sur cette ressource. Cette politique accorde également un accès complet à toutes les CloudFront actions Amazon.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
        "Effect": "Allow",
        "Action": "cloudfront:*",
        "Resource": [
            "arn:aws:cloudfront:*",
```

```
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ]
        }
    ]
}
```

Cette politique fournit des autorisations pour toutes les CloudFront actions. Mais comme elle définit la ressource S3 `amzn-s3-demo-bucket` sans définir d'actions S3, le récapitulatif de politique inclut l'avertissement suivant :

`This policy defines some actions, resources, or conditions that do not provide permissions. To grant access, policies must have an action that has an applicable resource or condition.`

Pour corriger cette politique de façon à fournir des autorisations de compartiment S3, vous devez définir des actions S3 pouvant être exécutées sur une ressource du compartiment.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloudfront:*",
                "s3:CreateBucket",
                "s3:ListBucket*",
                "s3:PutBucket*",
                "s3:GetBucket*"
            ],
            "Resource": [
                "arn:aws:cloudfront:*",
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        }
    ]
}
```

Sinon, pour corriger cette politique afin de ne fournir que des CloudFront autorisations, supprimez la ressource S3.

Une condition est définie sans action applicable

La politique ci-dessous définit deux actions Amazon S3 pour toutes les ressources S3, si le préfixe S3 est égal à `custom` et l'ID de version à `1234`. Toutefois, la clé de condition `s3:VersionId` est utilisée pour le balisage de version d'objet et n'est pas pris en charge par les actions de compartiment définies. Pour savoir quelles conditions sont prises en charge par une action, voir Actions, ressources et clés de condition pour les AWS services et choisissez le service pour consulter la documentation de service relative aux clés de condition.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
}
```

Cette politique fournit les autorisations pour l'action `s3:ListBucketVersions` et l'action `s3:ListBucket` si le nom du compartiment inclut le préfixe `custom`. Mais comme la condition

`s3:VersionId` n'est prise en charge par aucune des actions définies, le récapitulatif de politique inclut l'erreur suivante :

```
This policy does not grant any permissions. To grant access, policies must have an action that has an applicable resource or condition.
```

Pour corriger cette politique de façon à utiliser le balisage de version d'objet S3, vous devez définir une action S3 qui prend en charge la clé de condition `s3:VersionId`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket",
                "s3:GetObjectVersion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
}
```

Cette politique fournit des autorisations pour chaque action et condition de la politique. Toutefois, la politique ne fournit toujours pas d'autorisations, car il n'existe pas de cas où une même action correspond à deux conditions. À la place, vous devez créer deux instructions séparées : chacune comprendra uniquement les actions et les conditions auxquelles elles s'appliquent.

Pour corriger cette politique, créez deux instructions. La première instruction inclut les actions qui prennent en charge la condition `s3:prefix`, et la seconde instruction inclut les actions qui prennent en charge la condition `s3:VersionId`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": "custom"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:GetObjectVersion",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:VersionId": "1234"
                }
            }
        }
    ]
}
```

Résolution des problèmes de gestion des politiques

Vous pouvez diagnostiquer et résoudre les problèmes liés à la gestion des politiques.

Attacher ou détacher une politique dans un compte IAM

Certaines politiques AWS gérées sont liées à un service. Ces politiques sont utilisées uniquement avec un rôle lié à un service pour ce service. Dans la console IAM, lorsque vous affichez la page Détails de la politique, celle-ci contient une bannière signalant que la politique est liée à un service. Vous ne pouvez pas attacher cette politique à un utilisateur, un groupe ou un rôle au sein d'IAM. Lorsque vous créez un rôle lié à un service pour le service, cette politique est automatiquement attachée à votre nouveau rôle. Puisque la politique est obligatoire, vous ne pouvez pas la détacher du rôle lié au service.

Modification des politiques pour vos identités IAM en fonction de leur activité

Vous pouvez mettre à jour les politiques pour vos identités IAM (utilisateurs, groupes et rôles) en fonction de leur activité. Pour ce faire, consultez les événements de votre compte dans CloudTrail l'historique des événements. CloudTrail les journaux d'événements contiennent des informations détaillées sur les événements que vous pouvez utiliser pour modifier les autorisations de la politique.

Un utilisateur ou un rôle tente d'effectuer une action AWS et cette demande est refusée.

> Déterminez si l'utilisateur ou le rôle doit avoir l'autorisation d'effectuer l'action. Si tel est le cas, vous pouvez ajouter l'action et même l'ARN de la ressource à laquelle il a tenté d'accéder avec sa politique.

Un utilisateur ou un rôle dispose d'autorisations qu'il n'utilise pas.

> Envisagez de supprimer ces autorisations de leur politique. Veillez à ce que vos politiques accordent le privilège le plus faible requis pour exécuter uniquement les actions nécessaires.

Pour plus d'informations sur l'utilisation CloudTrail, consultez la section Affichage CloudTrail des événements dans la CloudTrail console dans le Guide de AWS CloudTrail l'utilisateur.

Résolution des problèmes de documents de politique JSON

Vous pouvez diagnostiquer et résoudre les problèmes liés aux documents de politique JSON.

Valider vos politiques

Lorsque vous créez ou modifiez une politique JSON, IAM peut effectuer une validation de politique pour vous aider à créer une politique efficace. IAM identifie les erreurs de syntaxe JSON, tandis que IAM Access Analyzer fournit des vérifications de politique supplémentaires avec des

recommandations pour vous aider à affiner vos politiques. Pour en savoir plus sur la validation de politiques, veuillez consulter Validation de politique IAM. Pour en savoir plus sur les vérifications des politiques IAM Access Analyzer et les recommandations exploitables, veuillez consulter Validation de politique IAM Access Analyzer.

Je n'ai pas d'autorisations pour la validation de politique dans l'éditeur JSON

Dans le Console de gestion AWS, l'erreur suivante peut s'afficher si vous n'êtes pas autorisé à consulter les résultats de validation des politiques d'IAM Access Analyzer :

```
You need permissions. You do not have the permissions required to perform this operation. Ask your administrator to add permissions.
```

Pour corriger cette erreur, demandez à votre administrateur d'ajouter l'autorisation `access-analyzer:ValidatePolicy` pour vous.

Plusieurs objets de politique JSON

Une politique IAM doit inclure un seul objet JSON. Vous désignez un objet en le plaçant entre accolades { }. Vous pouvez imbriquer d'autres objets dans un objet JSON en intégrant des accolades { } supplémentaires dans la paire extérieure. Une politique ne doit contenir qu'une seule paire d'accolades { } la plus externe. L'exemple suivant est incorrect car il contient deux objets au niveau supérieur (appelés en*red*) :

JSON

```
{
      "Version":"2012-10-17",
      "Statement":
      {
         "Effect":"Allow",
         "Action":"ec2:Describe*",
         "Resource":"*"
      }
   }
   {
      "Statement": {
         "Effect": "Allow",
         "Action": "s3:*",
         "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
      }
```

```
    }
```

Toutefois, il est possible de rectifier l'exemple précédent à l'aide d'une syntaxe de politique appropriée. Au lieu d'utiliser deux objets de politique complets, avec chacun son propre élément `Statement`, vous pouvez combiner les deux blocs en un seul élément `Statement`. La valeur de l'élément `Statement` est un tableau de deux objets, comme illustré dans l'exemple suivant (marqué en gras) :

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":"*"
        },
        {
          "Effect": "Allow",
          "Action": "s3:*",
          "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
      ]
    }
```

Plusieurs éléments de instruction JSON

Au premier abord, cette erreur peut sembler être une variante de celle de la section précédente. Toutefois, d'un point de vue syntaxique, il s'agit d'un type d'erreur différent. L'exemple suivant comporte un seul objet de politique, comme indiqué par la paire de parenthèses { } au premier niveau. Toutefois, cet objet contient deux éléments `Statement`.

Une politique IAM doit comporter un seul élément `Statement`, composé du nom (`Statement`) affiché à gauche d'un signe deux-points et suivi de sa valeur à droite. La valeur d'un élément `Statement` doit être un objet, indiqué par des accolades { }, contenant un élément `Effect`, un élément `Action` et un élément `Resource`. L'exemple suivant est incorrect car il contient deux `Statement` éléments dans l'objet de politique (appelé dans*red*) :

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
      },
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
      }
    }
```

Un objet de valeur peut être un tableau de plusieurs objets de valeur. Pour résoudre ce problème, combinez les deux éléments `Statement` en un même élément avec un tableau d'objets, comme illustré dans l'exemple suivant (marqué en gras) :

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":"*"
        },
        {
          "Effect": "Allow",
          "Action": "s3:*",
          "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
     ]
    }
```

La valeur de l'élément `Statement` est un tableau d'objets. Dans cet exemple, le tableau se compose de deux objets, chaque objet étant à lui seul une valeur correcte pour un élément `Statement`. Chaque objet du tableau est séparé par des virgules.

Plusieurs éléments Effect, Action ou Resource dans un élément d'instruction JSON

Du côté des valeurs de la `Statement` name/value paire, l'objet doit être composé d'un seul `Effect` élément, d'un `Action` élément et d'un seul `Resource` élément. La politique suivante est incorrecte, car elle comporte deux éléments `Effect` dans la `Statement` :

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Effect": "Allow",
        "Action": "ec2:* ",
        "Resource": "*"
      }
    }
```

> **Note**
>
> Le moteur de politiques n'autorise pas ce type d'erreurs dans les politiques nouvelles ou modifiées. Il continue toutefois à autoriser les politiques enregistrées avant sa mise à jour. Les politiques existantes contenant l'erreur se comportent comme suit :
>
> - Plusieurs éléments `Effect` : seul le dernier élément `Effect` est pris en compte. Les autres sont ignorés.
> - Plusieurs éléments `Action` : tous les éléments `Action` sont combinés en interne et traités comme s'il s'agissait d'une seule liste.
> - Plusieurs éléments `Resource` : tous les éléments `Resource` sont combinés en interne et traités comme s'il s'agissait d'une seule liste.

> Le moteur de politique ne vous permet pas d'enregistrer une politique contenant des erreurs syntaxiques. Corrigez les erreurs figurant dans la politique avant d'enregistrer. Passez en revue et corrigez toutes les recommandations de validation de politique pour vos politiques.

Dans chaque cas, la solution consiste à supprimer l'élément supplémentaire incorrect. Pour les éléments `Effect`, cela est très facile : si vous voulez que l'exemple précédent refuse des autorisations aux instances Amazon EC2, il vous suffit de supprimer la ligne `"Effect": "Allow",` de la politique, comme suit :

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Action": "ec2:*",
        "Resource": "*"
      }
    }
```

Toutefois, lorsque l'élément dupliqué est `Action` ou `Resource`, la résolution peut s'avérer plus complexe. Vous voulez peut-être autoriser (ou refuser) des autorisations pour plusieurs actions ou vous souhaitez contrôler l'accès à plusieurs ressources. Par exemple, l'exemple suivant est incorrect car il comporte plusieurs `Resource` éléments (appelés en*red*) :

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
      }
```

```
    }
```

Chacun des éléments requis dans l'objet de valeur d'un élément `Statement` ne doit y figurer qu'une seule fois. La solution consiste à placer chaque valeur dans un tableau. L'exemple suivant illustre cette méthode : les deux éléments Resource sont inclus dans un même élément `Resource` avec un tableau comme objet de valeur (marqué en gras) :

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": [
          "arn:aws:s3:::amzn-s3-demo-bucket",
          "arn:aws:s3:::amzn-s3-demo-bucket/*"
        ]
      }
    }
```

Élément de version JSON manquant

Un élément de politique `Version` est différent d'une version de politique. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. En comparaison, une version de politique est créée lorsque vous apportez des modifications à une politique gérée par le client dans IAM. La politique modifiée ne remplace pas la politique existante. À la place, IAM crée une nouvelle version de la politique gérée. Pour en savoir plus sur l'élément de politique `Version`, consultez Éléments de politique JSON IAM : Version. Pour en savoir plus sur les versions de politiques, consultez the section called "Gestion des versions des politiques IAM".

À mesure que les AWS fonctionnalités évoluent, de nouvelles fonctionnalités sont ajoutées aux politiques IAM pour prendre en charge ces fonctionnalités. Parfois, la mise à jour de la syntaxe de politique inclut un nouveau numéro de version. Si vous utilisez les nouvelles fonctions de la syntaxe de politique dans votre politique, vous devez indiquer le numéro de la version utilisée au moteur d'analyse de politique.

Le numéro de version de politique par défaut est « 2008-10-17 ». Si vous voulez utiliser une fonction de politique plus récente, vous devez spécifier le numéro de version prenant en charge cette fonction. Nous vous recommandons de toujours inclure le numéro de version de syntaxe de politique le plus récent, à savoir actuellement `"Version": "2012-10-17"`.

Par exemple, la politique suivante est incorrecte, car elle utilise une variable de politique `${...}` dans l'ARN d'une ressource. Mais il ne parvient pas à spécifier une version de syntaxe de stratégie prenant en charge les variables de stratégie (appelées dans*red*) :

```
{
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

L'ajout d'un élément `Version` au début de la politique avec la valeur `2012-10-17`, la première version de l'API IAM qui prend en charge les variables de politique, permet de corriger ce problème (marqué en gras) :

JSON

```
{
  "Version":"2012-10-17",
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

Résolution des problèmes liés aux clés d’accès et aux clés de sécurité FIDO

Utilisez ces informations pour identifier et résoudre les problèmes courants que vous pouvez rencontrer lors de l'utilisation des clés de sécurité FIDO2.

Rubriques

- Je ne peux pas activer ma clé de sécurité FIDO
- Je ne peux pas me connecter à l'aide de ma clé de sécurité FIDO
- J'ai perdu ou cassé ma clé de sécurité FIDO
- Autres problèmes

Je ne peux pas activer ma clé de sécurité FIDO

Consultez les solutions suivantes en fonction de votre statut (utilisateur IAM ou administrateur système).

Utilisateurs IAM

Si vous ne pouvez pas activer votre clé de sécurité FIDO, vérifiez les points suivants :

- Utilisez-vous une configuration prise en charge ?

  IAM prend en charge les dispositifs de sécurité FIDO2 qui se connectent à vos dispositifs via USB, Bluetooth ou NFC. IAM prend également en charge les authentificateurs de plateforme comme TouchID ou FaceID. IAM ne prend pas en charge l’enregistrement de clés d’accès locales pour Windows Hello. Pour créer et utiliser des clés d’accès, les utilisateurs de Windows doivent utiliser l’authentification entre appareils, qui consiste à utiliser une clé d’accès provenant d’un appareil, comme un appareil mobile, ou une clé de sécurité matérielle pour se connecter sur un autre appareil, tel qu’un ordinateur portable.

  Pour plus d'informations sur les appareils et les navigateurs que vous pouvez utiliser avec WebAuthn et AWS, consultezConfigurations prises en charge pour l’utilisation des clés d’accès et des clés de sécurité.
- Utilisez-vous des plug-ins de navigateur ?

AWS ne prend pas en charge l'utilisation de plugins pour ajouter la prise en charge des WebAuthn navigateurs. Utilisez plutôt un navigateur qui offre une prise en charge native de la WebAuthn norme.

Même si vous utilisez un navigateur compatible, il se peut que vous ayez un plugin incompatible avec WebAuthn. Un plug-in incompatible peut vous empêcher d'activer et d'utiliser votre clé FIDO-compliant de sécurité. Désactivez les plug-ins qui pourraient être incompatibles et redémarrer votre navigateur. Réessayez ensuite d’activer la clé de sécurité FIDO.

- Disposez-vous des autorisations appropriées ?

  Si vous ne rencontrez pas l'un des problèmes de compatibilité ci-dessus, vous n'avez peut-être pas les autorisations appropriées. Contactez votre administrateur système.

Administrateurs système

Si vos utilisateurs IAM ne peuvent pas activer leurs clés de sécurité FIDO bien qu’ils utilisent une configuration prise en charge, vérifiez leurs autorisations. Pour voir un exemple détaillé, veuillez consulter Didacticiel IAM : permettre aux utilisateurs de gérer leurs informations d'identification et leurs paramètres MFA.

Je ne peux pas me connecter à l'aide de ma clé de sécurité FIDO

Si vous ne parvenez pas à vous connecter à l' Console de gestion AWS aide de votre clé de sécurité FIDO, consultez Configurations prises en charge pour l’utilisation des clés d’accès et des clés de sécurité d'abord. Si vous utilisez une configuration prise en charge mais que vous ne pouvez pas vous connecter, contactez votre administrateur système pour obtenir de l'aide.

J'ai perdu ou cassé ma clé de sécurité FIDO

Jusqu'à huit dispositifs MFA de n'importe quelle combinaison de types MFA actuellement pris en charge peuvent être attribués à un utilisateur. Avec plusieurs dispositifs MFA, vous n'en avez besoin que d'un seul pour vous connecter à la Console de gestion AWS. Le remplacement d'une clé de sécurité FIDO est semblable à celui d'un jeton TOTP matériel. En cas de perte ou de casse d’un appareil MFA, consultez Restauration d’une identité protégée par MFA dans IAM.

Autres problèmes

Si vous rencontrez un problème avec les clés de sécurité FIDO qui n'est pas traité ici, effectuez l'une des actions suivantes :

- Utilisateurs IAM : contactez votre administrateur système.
- Compte AWS utilisateurs root : contactez le AWS support.

Résolution des problèmes liés aux rôles IAM

Utilisez ces informations pour identifier et résoudre les problèmes courants que vous pouvez rencontrer lors de l'utilisation des rôles IAM.

Rubriques

- Je ne parviens pas à endosser un rôle
- Un nouveau rôle est apparu dans mon AWS compte
- Je ne parviens pas à modifier ou à supprimer un rôle dans mon Compte AWS
- Je ne suis pas autorisé à jouer : iam : PassRole
- Pourquoi ne puis-je pas assumer un rôle dans le cadre d'une session de 12 heures ? (AWS CLI, AWS API)
- Je reçois une erreur lorsque j'essaie de changer de rôle dans la console IAM
- Mon rôle dispose d'une politique qui me permet d'effectuer une action, mais j'obtiens « Accès refusé »
- Le service n'a pas créé la version de politique par défaut du rôle
- Il n'existe aucun cas d'utilisation pour un rôle de service dans la console

Je ne parviens pas à endosser un rôle

Vérifiez les éléments suivants :

- Pour permettre aux utilisateurs d'assumer à nouveau le rôle actuel au cours d'une session de rôle, spécifiez l'ARN du rôle ou l' Compte AWS ARN en tant que principal dans la politique de confiance des rôles. Services AWS qui fournissent des ressources de calcul telles qu'Amazon EC2, Amazon ECS, Amazon EKS et Lambda fournissent des informations d'identification temporaires et mettent automatiquement à jour ces informations d'identification. Cela garantit que vous disposez toujours

d'un ensemble d'informations d'identification valide. Pour ces services, il n'est pas nécessaire d'endosser à nouveau le rôle actuel pour obtenir des informations d'identification temporaires. Toutefois, si vous avez l'intention de transférer des balises de session ou une politique de session, vous devez endosser à nouveau le rôle actuel. Pour savoir comment modifier une politique de confiance des rôles afin d'ajouter l'ARN ou Compte AWS l'ARN du rôle principal, consultezMise à jour d'une politique d'approbation de rôle.

- Lorsque vous assumez un rôle à l'aide de Console de gestion AWS, veillez à utiliser le nom exact de votre rôle. Les noms de rôle sont sensibles à la casse lorsque vous endossez un rôle.
- Lorsque vous assumez un rôle à l'aide de l' AWS STS API ou AWS CLI, assurez-vous d'utiliser le nom exact de votre rôle dans l'ARN. Les noms de rôle sont sensibles à la casse lorsque vous endossez un rôle.
- Lorsque vous assumez un rôle à l'aide d'un fournisseur d'identité SAML-based fédéré et que le chiffrement SAML est activé, assurez-vous d'avoir chargé une clé de déchiffrement privée valide pour le fournisseur d'identité SAML. Pour de plus amples informations, veuillez consulter Gestion des clés de chiffrement SAML.
- Vérifiez que votre politique IAM vous accorde l'autorisation d'appeler `sts:AssumeRole` pour le rôle que vous voulez endosser. L'élément `Action` de votre politique IAM doit vous autoriser à appeler l'action `AssumeRole`. En outre, l'élément `Resource` de votre politique IAM doit spécifier le rôle que vous voulez endosser. Par exemple, l'élément `Resource` peut spécifier un rôle en utilisant son ARN (Amazon Resource Name) ou à l'aide d'un caractère générique (*). Par exemple, au moins une politique s'appliquant à vous doit accorder des autorisations similaires à ce qui suit :

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
```

- Vérifiez que votre identité IAM est balisée avec les balises que la politique IAM requiert. Par exemple, dans la politique d'autorisations suivante, l'élément `Condition` exige que vous, en tant que principal demandant à endosser le rôle, ayez une balise spécifique. Vous devez être balisé avec `department = HR` ou `department = CS`. Dans le cas contraire, vous ne pouvez pas endosser le rôle. Pour en savoir plus sur le balisage des utilisateurs et rôles IAM, veuillez consulter the section called "Balises pour les ressources IAM".

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "*",
    "Condition": {"StringEquals": {"aws:PrincipalTag/department": [
```

```
            "HR",
            "CS"
        ]}}
```

- Assurez-vous que vous respectez toutes les conditions spécifiées dans la politique d'approbation du rôle. Une `Condition` peut spécifier une date d'expiration, un ID externe, ou exiger qu'une demande provienne uniquement d'adresses IP spécifiques. Prenons l'exemple suivant. Si la date actuelle se situe après la date spécifiée, la politique ne correspond jamais et, par conséquent, elle ne peut par vous accorder l'autorisation d'endosser le rôle.

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
    "Condition": {
        "DateLessThan" : {
            "aws:CurrentTime" : "2016-05-01T12:00:00Z"
        }
    }
```

- Vérifiez que l'entité Compte AWS à partir de laquelle vous appelez `AssumeRole` est une entité approuvée pour le rôle que vous assumez. Les entités approuvées sont définies en tant que `Principal` dans la politique d'approbation d'un rôle. L'exemple suivant est une politique de confiance attachée au rôle que vous voulez endosser. Dans cet exemple, l'ID de compte avec l'utilisateur IAM avec lequel vous vous êtes connecté doit être 123456789012. Si votre numéro de compte n'est pas répertorié dans l'élément `Principal` de la politique de confiance du rôle, vous ne pouvez pas endosser le rôle. Peu importent les autorisations qui vous sont accordées dans les politiques d'accès. Notez que l'exemple de politique limite les autorisations aux actions susceptibles de se produire entre le 1er juillet 2017 et le 31 décembre 2017 (UTC), inclus. Si vous vous connectez avant ou après ces dates, la politique de correspond pas et vous ne pouvez pas endosser le rôle.

```
    "Effect": "Allow",
    "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
    "Action": "sts:AssumeRole",
    "Condition": {
      "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
      "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
    }
```

- Identité de la source : les administrateurs peuvent configurer des rôles pour exiger que les identités transmettent une chaîne personnalisée identifiant la personne ou l'application qui effectue des actions AWS, appelée identité source. Vérifiez si le rôle endossé exige qu'une identité source soit définie. Pour de plus amples informations sur l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.

Un nouveau rôle est apparu dans mon AWS compte

Certains AWS services nécessitent que vous utilisiez un type unique de rôle de service directement lié au service. Ce rôle lié à un service est prédéfini par le service et comprend toutes les autorisations nécessaires au service. La configuration d'un service est ainsi simplifiée, étant donné que vous n'avez pas besoin d'ajouter manuellement les autorisations requises. Pour obtenir des informations générales sur les rôles liés à un service, veuillez consulter Créer un rôle lié à un service.

Vous utilisez peut être déjà un service lorsqu'il commence à prendre en charge des rôles de service liés à un service. Dans ce cas, il est possible que vous receviez un e-mail vous informant d'un nouveau rôle dans votre compte. Ce rôle comprend toutes les autorisations dont le service a besoin pour effectuer des actions en votre nom. Aucune action de votre part n'est requise pour prendre ce rôle en charge. Cependant, vous ne devez pas supprimer le rôle de votre compte. En effet, la suppression risquerait de supprimer des autorisations dont le service a besoin pour accéder aux ressources AWS . Vous pouvez consulter les rôles liés aux services dans votre compte en accédant à la page Rôles IAM de la console IAM. Service-linked les rôles apparaissent avec (Service-linked rôle) dans la colonne Entités fiables du tableau.

Pour plus d'informations sur les services qui prennent en charge les rôles liés aux services, consultez AWS services qui fonctionnent avec IAM et recherchez les services dont la valeur est Oui dans la colonne Service-Linked Rôle. Pour plus d'informations sur l'utilisation du rôle lié à un service, sélectionnez le lien Oui.

Je ne parviens pas à modifier ou à supprimer un rôle dans mon Compte AWS

Vous ne pouvez pas supprimer ni modifier les autorisations d'un rôle lié à un service dans IAM. Ces rôles comportent des approbations et des autorisations prédéfinies requises par le service afin d'exécuter des actions en votre nom. Vous pouvez utiliser la console IAM ou l' AWS CLI API pour modifier uniquement la description d'un rôle lié à un service. Vous pouvez consulter les rôles liés aux services dans votre compte en accédant à la page Rôles IAM de la console. Service-linked les rôles

apparaissent avec (Service-linked rôle) dans la colonne Entités fiables du tableau. Une bannière sur la page Récapitulatif d'un rôle indique aussi que le rôle est lié à un service. Vous pouvez gérer et supprimer ces rôles uniquement via le service lié, si celui-ci prend en charge l'action. Soyez vigilant lorsque vous modifiez ou supprimez un rôle lié à un service car cela peut supprimer des autorisations requises par le service pour accéder aux ressources AWS .

Pour plus d'informations sur les services qui prennent en charge les rôles liés aux services, consultez AWS services qui fonctionnent avec IAM et recherchez les services dont la valeur est Oui dans la colonne Service-Linked Rôle.

Je ne suis pas autorisé à jouer : iam : PassRole

Lorsque vous créez un rôle lié à un service, vous devez être autorisé à transmettre ce rôle au service. Certains services créent automatiquement un rôle lié à un service dans votre compte lorsque vous effectuez une action dans ce service. Par exemple, Amazon EC2 Auto Scaling crée automatiquement le rôle lié au service `AWSServiceRoleForAutoScaling` la première fois que vous créez un groupe Auto Scaling. Si vous essayez de créer un groupe Auto Scaling sans l'autorisation `PassRole`, vous recevez le message d'erreur suivant :

```
ClientError: An error occurred (AccessDenied) when calling the
PutLifecycleHook operation: User: arn:aws:sts::111122223333:assumed-role/
Testrole/Diego is not authorized to perform: iam:PassRole on resource:
arn:aws:iam::111122223333:role/aws-service-role/autoscaling.amazonaws.com/
AWSServiceRoleForAutoScaling
```

Pour corriger cette erreur, demandez à votre administrateur d'ajouter l'autorisation `iam:PassRole` pour vous.

Pour savoir quels services prennent en charge les rôles liés à un service, veuillez consulter AWS services qui fonctionnent avec IAM. Pour savoir si un service crée automatiquement un rôle lié à un service, sélectionnez le lien Oui pour afficher la documentation du rôle lié à ce service.

Pourquoi ne puis-je pas assumer un rôle dans le cadre d'une session de 12 heures ? (AWS CLI, AWS API)

Lorsque vous utilisez les opérations de l' AWS STS `AssumeRole*`API ou de la `assume-role*` CLI pour assumer un rôle, vous pouvez spécifier une valeur pour le `DurationSeconds` paramètre. Vous pouvez spécifier une valeur comprise entre 900 secondes (15 minutes) et la valeur de durée

de session maximale définie pour le rôle. Si vous spécifiez une valeur supérieure à ce paramètre, l'opération échoue. La valeur maximale de ce paramètre est de 12 heures. Par exemple, si vous spécifiez une durée de session de 12 heures, mais que votre administrateur a défini la durée de session maximale à 6 heures, votre opération échoue. Pour savoir comment afficher la valeur maximale pour votre rôle, veuillez consulter Mettre à jour la durée de session maximale pour un rôle.

Si vous utilisez la création de chaînes de rôles (en utilisant un rôle pour endosser un second rôle), votre session est limitée à une heure maximum. Si vous utilisez ensuite le paramètre `DurationSeconds` pour fournir une valeur supérieure à une heure, l'opération échoue.

Je reçois une erreur lorsque j'essaie de changer de rôle dans la console IAM

Les informations que vous entrez sur la page Changer de rôle doivent correspondre à celles du rôle. Sinon, l'opération échoue et vous recevez l'erreur suivante :

```
Invalid information in one or more fields. Check your information or
contact your administrator.
```

Si vous recevez cette erreur, confirmez que les informations suivantes sont correctes :

- ID de compte ou alias : l' Compte AWS identifiant est un numéro à 12 chiffres. Votre compte possède peut-être un alias, qui est un identifiant convivial, tel que le nom de votre entreprise, qui peut être utilisé à la place de votre Compte AWS identifiant. Vous pouvez utiliser l'ID de compte ou l'alias dans ce champ.
- Role name (Nom de rôle) : les noms de rôle sont sensibles à la casse. L'ID du compte et le nom du rôle doivent correspondre à ce qui est configuré pour le rôle.

Si vous continuez à recevoir un message d'erreur, contactez votre administrateur pour vérifier les informations précédentes. La politique d'approbation de rôle ou la politique utilisateur IAM peut limiter votre accès. Votre administrateur peut vérifier les autorisations pour ces politiques.

Mon rôle dispose d'une politique qui me permet d'effectuer une action, mais j'obtiens « Accès refusé »

Votre session peut être limitée par les politiques de session. Lorsque vous demandez des informations d'identification de sécurité temporaires par programmation à l'aide de AWS STS, vous pouvez éventuellement transmettre des politiques de session en ligne ou gérées. Les politiques

de session sont des politiques avancées que vous transmettez en tant que paramètre lorsque vous créez par programmation une session d'informations d'identification temporaires pour un rôle. Vous pouvez transmettre un seul document de politique de session en ligne JSON à l'aide du paramètre `Policy`. Vous pouvez utiliser le paramètre `PolicyArns` pour spécifier jusqu'à 10 politiques de session gérées. Les autorisations de la session obtenues sont une combinaison des politiques basées sur l'identité du rôle et des politiques de session. Sinon, si votre administrateur ou un programme personnalisé vous fournit des informations d'identification temporaires, ils peuvent avoir inclus une politique de session pour limiter votre accès.

Le service n'a pas créé la version de politique par défaut du rôle

Un rôle de service est un rôle qu'un service endosse pour effectuer des actions dans votre compte en votre nom. Lorsque vous configurez certains environnements de AWS service, vous devez définir un rôle que le service doit assumer. Dans certains cas, le service crée le rôle de service et sa politique dans IAM pour vous. Bien que vous puissiez modifier ou supprimer le rôle de service et sa politique depuis IAM, AWS ne le recommande pas. Le rôle et la politique sont destinés à être utilisés uniquement par ce service. Si vous modifiez la politique et configurez un autre environnement, lorsque le service tente d'utiliser le même rôle et la même politique, l'opération peut échouer.

Par exemple, lorsque vous utilisez AWS CodeBuild pour la première fois, le service crée un rôle nommé`codebuild-RWBCore-service-role`. Ce rôle de service utilise la politique nommée `codebuild-RWBCore-managed-policy`. Si vous modifiez la politique, elle crée une nouvelle version et enregistre cette version en tant que version par défaut. Si vous effectuez une opération ultérieure dans AWS CodeBuild, le service peut essayer de mettre à jour la politique. Si c'est le cas, vous recevez l'erreur suivante :

```
codebuild.amazon.com did not create the default version (V2) of the
codebuild-RWBCore-managed-policy policy that is attached to the codebuild-
RWBCore-service-role role. To continue, detach the policy from any other
identities and then delete the policy and the role.
```

Si vous recevez cette erreur, vous devez apporter des modifications dans IAM avant de pouvoir poursuivre votre opération de service. Tout d'abord, définissez la version de la politique par défaut sur V1 et réessayez l'opération. Si V1 a déjà été supprimé, ou si le choix de V1 ne fonctionne pas, nettoyez et supprimez la politique et le rôle existants.

Pour de plus amples informations sur la modification des politiques gérées, veuillez consulter Modification de politiques gérées par le client (console). Pour plus d'informations sur les versions de politique, veuillez consulter Gestion des versions des politiques IAM.

Pour supprimer un rôle de service et sa politique

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation, sélectionnez Policies (Politiques).
3. Dans la liste des politiques, sélectionnez le nom de la politique à supprimer.
4. Choisissez l'onglet Entités attachées pour afficher les utilisateurs, groupes ou rôles IAM qui utilisent cette politique. Si l'une de ces identités utilise la politique, effectuez les tâches suivantes :

    a. Créez une nouvelle politique gérée avec les autorisations nécessaires. Pour vous assurer que les identités disposent des mêmes autorisations avant et après vos actions, copiez le document de politique JSON à partir de la politique existante. Créez ensuite la nouvelle politique gérée et collez le document JSON comme décrit dans la section Création de politiques à l'aide de l'éditeur JSON.

    b. Pour chaque identité affectée, joignez la nouvelle politique, puis détachez l'ancienne. Pour de plus amples informations, veuillez consulter Ajout et suppression d'autorisations basées sur l'identité IAM.
5. Dans le panneau de navigation, sélectionnez Roles (Rôles).
6. Dans la liste des rôles, sélectionnez le nom du rôle à supprimer.
7. Cliquez sur l'onglet Relations d'approbation pour afficher les entités qui peuvent endosser le rôle. Si une entité autre que le service est répertoriée, effectuez les tâches suivantes :

    a. Créez un nouveau rôle qui approuve ces entités.

    b. Politique que vous avez créée à l'étape précédente. Si vous avez ignoré cette étape, créez la nouvelle politique gérée maintenant.

    c. Avisez toute personne qui endossait le rôle qu'elle ne peut plus le faire. Donnez-lui des informations sur la façon d'endosser le nouveau rôle et d'avoir les mêmes autorisations.
8. Supprimez la politique.
9. Supprimez le rôle.

Il n'existe aucun cas d'utilisation pour un rôle de service dans la console

Certains services exigent que vous créiez manuellement un rôle de service pour accorder au service les autorisations nécessaires pour effectuer des actions en votre nom. Si le service n'est

pas répertorié dans la console IAM, vous devez répertorier manuellement le service en tant que principal de confiance. Si la documentation du service ou de la fonctionnalité que vous utilisez ne contient pas d'instructions pour intégrer le service à la liste des principaux de confiance, soumettez des commentaires pour la page.

Pour créer manuellement un rôle de service, vous devez connaître le principal de service du service qui endossera le rôle. Un principal de service est un identifiant utilisé pour accorder des autorisations à un service. Le principal de service est défini par le service.

Vous pouvez trouver le principal de service pour certains services en vérifiant les éléments suivants :

1. Ouvrir AWS services qui fonctionnent avec IAM.
2. Vérifiez si le service a la valeur Oui dans la colonne Service-linked des rôles.
3. Choisissez le lien Oui pour consulter la documentation du rôle lié à ce service.
4. Accédez à la section des autorisations de Service-linked rôle pour ce service pour afficher le principal du service.

Vous pouvez créer manuellement un rôle de service à l'aide de commandes AWS CLI ou d'opérations d'API AWS. Pour créer manuellement un rôle de service à l'aide de la console IAM, effectuez les tâches suivantes :

1. Créez un rôle IAM à l'aide de votre ID de compte. N'attachez pas de politique et n'accordez aucune autorisation. Pour en savoir plus, consultez Créer un rôle pour attribuer des autorisations à un utilisateur IAM.
2. Ouvrez le rôle et modifiez la relation d'approbation. Au lieu d'approuver le compte, le rôle doit approuver le service. Par exemple, mettez à jour l'élément `Principal` suivant :

   ```
   "Principal": { "AWS": "arn:aws:iam::123456789012:root" }
   ```

   Modifiez le principal par la valeur de votre service, par exemple, IAM.

   ```
   "Principal": { "Service": "iam.amazonaws.com" }
   ```

3. Ajoutez les autorisations requises par le service en attachant des politiques d'autorisations au rôle.
4. Revenez au service qui nécessite les autorisations et utilisez la méthode décrite dans la documentation pour informer le service du nouveau rôle de service.

Dépannage d'IAM et Amazon EC2

Les informations suivantes peuvent vous aider à résoudre les problèmes IAM liés à Amazon EC2.

Rubriques

- Lorsque j'essaie de lancer une instance, je ne vois pas le rôle dans la liste de Rôles IAM de la console Amazon EC2.
- Les informations d'identification sur mon instance concernent le mauvais rôle
- Quand je tente d'appeler AddRoleToInstanceProfile, je reçois un message d'erreur AccessDenied.
- Amazon EC2 : quand je tente de lancer l'instance avec un rôle, je reçois un message d'erreur AccessDenied
- Je ne parviens pas à accéder aux informations d'identification de sécurité temporaires sur mon instance EC2
- Que signifient les erreurs dans le document info de la sous-arborescence IAM ?

Lorsque j'essaie de lancer une instance, je ne vois pas le rôle dans la liste de Rôles IAM de la console Amazon EC2.

Vérifiez les éléments suivants :

- Si vous êtes connecté en tant qu'utilisateur IAM, vérifiez que vous avez l'autorisation d'appeler `ListInstanceProfiles`. Pour plus d'informations sur les autorisations nécessaires pour utiliser ces rôles, consultez Autorisations requises pour l'utilisation de rôles avec Amazon EC2. Pour de plus amples informations sur l'ajout d'autorisations à un utilisateur, veuillez consulter Gestion des politiques IAM.

  Si vous ne pouvez pas modifier vos propres autorisations, vous devez contacter un administrateur qui peut utiliser IAM pour mettre à jour vos autorisations.
- Si vous avez créé un rôle à l'aide de l'interface de ligne de commande ou l'API IAM, vérifiez les points suivants :
  - Vous avez créé un profil d'instance et ajouté le rôle à celui-ci.
  - Vous avez utilisé le même nom pour le rôle et le profil d'instance. Si vous nommez votre rôle et votre profil d'instance différemment, le nom correct du rôle ne s'affichera pas dans la console Amazon EC2.

La liste des Rôles IAM de la console Amazon EC2 répertorie les noms des profils d'instance, pas les noms des rôles. Vous devez sélectionner le nom du profil d'instance qui contient le rôle choisi. Pour plus informations sur les profils d'instance, veuillez consulter Utilisation des profils d'instance.

> **Note**
>
> Si vous utilisez la console IAM pour créer des rôles, vous n'avez pas besoin d'utiliser des profils d'instance. Pour chaque rôle que vous créez dans la console IAM, un profil d'instance est créé avec un nom identique à celui du rôle, et le rôle est ajouté automatiquement à ce profil d'instance. Un profil d'instance peut contenir un rôle IAM uniquement et cette limite ne peut pas être augmentée.

Les informations d'identification sur mon instance concernent le mauvais rôle

Le rôle dans le profil d'instance a peut-être été remplacé récemment. Si tel est le cas, votre application attendra la prochaine rotation des informations d'identification planifiée automatiquement pour que les informations d'identification de votre rôle deviennent disponibles.

Pour forcer la modification, vous devez dissocier le profil d'instance, puis associer le profil d'instance, ou vous pouvez arrêter votre instance, puis la redémarrer.

Quand je tente d'appeler **`AddRoleToInstanceProfile`**, je reçois un message d'erreur **`AccessDenied`**.

Si vous faites des demandes en tant qu'utilisateur IAM, vérifiez que vous avez les autorisation suivantes :

- `iam:AddRoleToInstanceProfile` avec la ressource correspondant à l'ARN du profil d'instance (par exemple, `arn:aws:iam::999999999999:instance-profile/ExampleInstanceProfile`).

Pour plus d'informations sur les autorisations nécessaires pour utiliser ces rôles, consultez Comment puis-je commencer ?. Pour de plus amples informations sur l'ajout d'autorisations à un utilisateur, veuillez consulter Gestion des politiques IAM.

Amazon EC2 : quand je tente de lancer l'instance avec un rôle, je reçois un message d'erreur **`AccessDenied`**

Vérifiez les éléments suivants :

- Lancez une instance sans profil d'instance. Cela vous permettra de vérifier que le problème se limite aux rôles IAM dans les instances Amazon EC2.
- Si vous faites des demandes en tant qu'utilisateur IAM, vérifiez que vous avez les autorisation suivantes :
  - `ec2:RunInstances` avec une ressource générique (« * »)
  - `iam:PassRole` avec la ressource correspondant à l'ARN du rôle (par exemple, `arn:aws:iam::999999999999:role/ExampleRoleName`)
- Appelez l'action IAM `GetInstanceProfile` pour vous assurer que vous utilisez un nom de profil d'instance valide ou un ARN de profil d'instance valide. Pour plus d'informations, veuillez consulter Utilisation de rôles IAM avec les instances Amazon EC2.
- Appelez l'action `GetInstanceProfile` IAM pour vous assurer que le profil d'instance dispose d'un rôle. Les profils d'instance vides échoueront et renverront l'erreur `AccessDenied`. Pour plus d'informations sur la création d'un rôle, veuillez consulter Création d'un rôle IAM.

Pour plus d'informations sur les autorisations nécessaires pour utiliser ces rôles, consultez Comment puis-je commencer ?. Pour de plus amples informations sur l'ajout d'autorisations à un utilisateur, veuillez consulter Gestion des politiques IAM.

Je ne parviens pas à accéder aux informations d'identification de sécurité temporaires sur mon instance EC2

Pour accéder aux informations d'identification de sécurité temporaires sur votre instance EC2, vous devez d'abord utiliser la console IAM pour créer un rôle. Ensuite, vous lancez une instance EC2 qui utilise ce rôle et vous examinez l'instance en cours d'exécution. Pour de plus amples informations, veuillez consulter Comment démarrer avec ce service ? dans Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2.

Si vous ne parvenez toujours pas à accéder à vos informations d'identification de sécurité temporaires sur votre instance EC2, vérifiez les points suivants :

- Pouvez-vous accéder à une autre partie du service des métadonnées d'instance (IMDS) ? Si ce n'est pas le cas, vérifiez qu'aucune règle de pare-feu ne bloque l'accès aux demandes envoyées à l'IMDS.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/
hostname; echo
```

- La sous-arborescence `iam` de l'IMDS existe-elle ? Si ce n'est pas le cas, vérifiez que votre instance dispose d'un profil d'instance IAM associé en appelant l'opération d'API EC2 `DescribeInstances` ou en utilisant la commande de la CLI `aws ec2 describe-instances`.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam;
 echo
```

- Consultez le document `info` dans la sous-arborescence IAM pour rechercher une erreur. S'il existe une erreur, veuillez consulter Que signifient les erreurs dans le document `info` de la sous-arborescence IAM ? pour plus d'informations.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam/
info; echo
```

Que signifient les erreurs dans le document **`info`** de la sous-arborescence IAM ?

Le document **`iam/info`** indique **`"Code":"InstanceProfileNotFound"`**

Votre profil d'instance IAM a été supprimé et Amazon EC2 ne peut plus fournir d'informations d'identification à votre instance. Vous devez attacher un profil d'instance valide à votre instance Amazon EC2.

S'il existe un profil d'instance avec le même nom, vérifiez que celui-ci n'a pas été supprimé et qu'un autre n'a pas été créé avec le même nom :

1. Appelez l'opération `GetInstanceProfile` IAM pour obtenir l'ID de profil d'instance `InstanceProfileId`.
2. Appelez l'opération `DescribeInstances` Amazon EC2 pour obtenir l'ID de profil d'instance `IamInstanceProfileId`.

3. Vérifiez que l'ID `InstanceProfileId` de l'opération IAM correspond à l'ID `IamInstanceProfileId` de l'opération Amazon EC2.

S' IDs ils sont différents, le profil d'instance attaché à vos instances n'est plus valide. Vous devez attacher un profil d'instance valide à l'instance.

Le document **`iam/info`** indique un succès, mais renvoie le message **`"Message":"Instance Profile does not contain a role..."`**

Le rôle a été supprimé du profil d'instance par l'action `RemoveRoleFromInstanceProfile` IAM. Vous pouvez utiliser l'action `AddRoleToInstanceProfile` IAM pour attacher un rôle au profil d'instance. Votre application devra attendre la prochaine actualisation planifiée pour accéder aux informations d'identification du rôle.

Pour forcer la modification, vous devez dissocier le profil d'instance, puis associer le profil d'instance, ou vous pouvez arrêter votre instance, puis la redémarrer.

Le document **`iam/security-credentials/[role-name]`** indique **`"Code":"AssumeRoleUnauthorizedAccess"`**

Amazon EC2 n'a pas l'autorisation d'endosser le rôle. L'autorisation d'endosser le rôle est contrôlée par la politique d'approbation attachée au rôle, comme dans l'exemple suivant. Utilisez l'API `UpdateAssumeRolePolicy` IAM pour mettre à jour la politique d'approbation.

JSON

```
{"Version":"2012-10-17",       "Statement": [{"Effect": "Allow","Principal":
 {"Service": ["ec2.amazonaws.com"]},"Action": ["sts:AssumeRole"]}]}
```

Votre application attendra jusqu'à la prochaine actualisation des informations d'identification planifiée automatiquement pour accéder aux informations d'identification du rôle.

Pour forcer la modification, vous devez dissocier le profil d'instance, puis associer le profil d'instance, ou vous pouvez arrêter votre instance, puis la redémarrer.

Résolution des problèmes liés à IAM et Amazon S3

Utilisez ces informations pour identifier et résoudre les problèmes courants que vous pouvez rencontrer lors de l'utilisation d'Amazon S3 et IAM.

Comment accorder un accès anonyme à un compartiment Amazon S3 ?

Vous utilisez une politique de compartiment Amazon S3 qui spécifie un caractère générique (*) dans l'élément `principal`, ce qui signifie que n'importe qui peut accéder au compartiment. Avec un accès anonyme, tout le monde (y compris les utilisateurs qui n'en ont pas Compte AWS) pourra accéder au bucket. Pour consulter un exemple de politique, veuillez consulter la rubrique Example Cases for Amazon S3 Bucket Policies (cas de politiques de compartiment Amazon S3) dans l'espace Amazon Simple Storage Service User Guide (guide de l'utilisateur du service simple de stockage Amazon).

Je suis connecté en tant que Compte AWS utilisateur root. Pourquoi est-ce que je ne parviens pas à accéder à un compartiment Amazon S3 dans mon compte ?

Dans certains cas, un utilisateur IAM peut disposer d'un accès complet à IAM et Amazon S3. Si l'utilisateur IAM affecte une politique de compartiment à un compartiment Amazon S3 sans spécifier l'utilisateur racine comme principal, l'utilisateur racine n'a pas accès au compartiment. Cependant, comme utilisateur racine, il est toujours possible d'accéder au compartiment. Pour ce faire, modifiez la politique de compartiment pour autoriser l'accès à l'utilisateur racine à partir de la console Amazon S3 ou de la AWS CLI. Utilisez le principal suivant, en le *123456789012* remplaçant par l'ID du Compte AWS.

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

Résolution des problèmes liés à la fédération SAML avec IAM

Utilisez ces informations pour identifier et résoudre les problèmes courants que vous pouvez rencontrer lors de l'utilisation de SAML 2.0 et de la fédération avec Gestion des identités et des accès AWS.

Rubriques

- Error: Your request included an invalid SAML response. To logout, click here.

- Erreur : RoleSessionName est obligatoire dans AuthnResponse (service : AWSSecurityTokenService ; code d'état : 400 ; code d'erreur : InvalidIdentityToken)
- Erreur : Non autorisé à exécuter sts : AssumeRoleWithSAML (service : AWSSecurityTokenService ; code d'état : 403 ; code d'erreur : AccessDenied)
- Erreur : RoleSessionName le champ AuthnResponse doit correspondre à [a-z A-Z _0-9+=, .@-] {2,64} (service : AWSSecurityTokenService ; code d'état : 400 ; code d'erreur :) InvalidIdentityToken
- Erreur : l'identité de la source doit correspondre à [a-z A-Z _0-9+=, .@-] {2,256} et ne pas commencer par « aws : » (service : ; code d'état : 400 AWSSecurityTokenService ; code d'erreur :) InvalidIdentityToken
- Erreur : signature de réponse non valide (service : AWSSecurityTokenService ; code d'état : 400 ; code d'erreur : InvalidIdentityToken)
- Erreur : clé privée non valide.
- Erreur : échec de la suppression de la clé privée.
- Erreur : impossible de supprimer la clé privée, car l'ID de la clé ne correspond pas à une clé privée.
- Erreur : échec de l'attribution du rôle : émetteur absent du fournisseur spécifié (service : AWSOpenIdDiscoveryService ; code d'état : 400 ; code d'erreur : AuthSamlInvalidSamlResponseException)
- Erreur : Impossible d'analyser les métadonnées.
- Erreur : impossible de mettre à jour le fournisseur d'identité. Aucune mise à jour n'est définie pour les métadonnées ou l'assertion de chiffrement.
- Erreur : impossible de définir le mode de chiffrement des assertions sur Obligatoire, car aucune clé privée n'est fournie.
- Erreur : impossible d'ajouter et de supprimer des clés privées dans la même requête. Définissez une valeur pour un seul des deux paramètres.
- Erreur : Le fournisseur spécifié n'existe pas.
- Erreur : la DurationSeconds valeur demandée dépasse la valeur MaxSessionDuration définie pour ce rôle.
- Erreur : la limite de deux clés privées est atteinte.
- Erreur : la réponse ne contient pas l'audience requise.

Error: Your request included an invalid SAML response. To logout, click here.

Cette erreur peut se produire lorsque la réponse SAML du fournisseur d'identité n'inclut pas d'attribut avec le `Name` défini sur `https://aws.amazon.com/SAML/Attributes/Role`. L'attribut doit contenir un ou plusieurs éléments `AttributeValue`, chacun contenant une paire de chaînes séparées par une virgule :

- L'ARN d'un rôle auquel l'utilisateur peut être mappé
- L'ARN du fournisseur SAML

L'erreur peut également se produire lorsque les valeurs d'attribut SAML envoyées par le fournisseur d'identité comportent un espace au début ou à la fin, ou d'autres caractères non valides dans les valeurs d'attribut SAML. Pour plus d'informations sur les valeurs attendues pour les attributs SAML, consultez Configuration d'assertions SAML pour la réponse d'authentification

Pour de plus amples informations, veuillez consulter Configuration d'assertions SAML pour la réponse d'authentification. Pour afficher la réponse SAML dans votre navigateur, suivez les étapes répertoriées dans la section Affichage d'une réponse SAML dans votre navigateur.

Erreur : RoleSessionName est obligatoire dans AuthnResponse (service : AWSSecurityTokenService ; code d'état : 400 ; code d'erreur : InvalidIdentityToken)

Cette erreur peut se produire lorsque la réponse SAML du fournisseur d'identité n'inclut pas d'attribut avec le `Name` défini sur `https://aws.amazon.com/SAML/Attributes/RoleSessionName`. La valeur de l'attribut est un identifiant pour l'utilisateur, généralement un ID utilisateur ou une adresse e-mail.

Pour plus d'informations, veuillez consulter Configuration d'assertions SAML pour la réponse d'authentification. Pour afficher la réponse SAML dans votre navigateur, suivez les étapes répertoriées dans la section Affichage d'une réponse SAML dans votre navigateur.

Erreur : Non autorisé à exécuter sts : AssumeRoleWithSAML (service : AWSSecurityTokenService ; code d'état : 403 ; code d'erreur : AccessDenied)

Cette erreur peut se produire si le rôle IAM spécifié dans la réponse SAML est mal orthographié ou n'existe pas. Assurez-vous d'utiliser le nom exact de votre rôle, car les noms de rôle sont sensibles à la casse. Corrigez le nom du rôle dans la configuration du fournisseur de service SAML.

Vous n'êtes autorisé à accéder que si votre politique d'approbation de rôle inclut l'action `sts:AssumeRoleWithSAML`. Si votre assertion SAML est configurée pour utiliser l'attribut `PrincipalTag`, votre politique d'approbation doit également inclure l'action `sts:TagSession`. Pour de plus amples informations sur les balises de session, veuillez consulter Transmettre les balises de session AWS STS.

Cette erreur peut se produire si vous n'avez pas d'autorisations `sts:SetSourceIdentity` dans votre politique de confiance de rôle. Si votre assertion SAML est configurée pour utiliser l'attribut `SourceIdentity`, votre politique d'approbation doit également inclure l'action `sts:SetSourceIdentity`. Pour de plus amples informations sur l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.

Cette erreur peut également se produire si un principal fédéré n'est pas autorisé à endosser le rôle. Le rôle doit disposer d'une politique d'approbation qui spécifie l'ARN du fournisseur d'identité SAML IAM en tant que principal (`Principal`). Le rôle contient également des conditions qui contrôlent les utilisateurs qui peuvent endosser le rôle. Vérifiez que les utilisateurs satisfont les exigences des conditions.

Cette erreur peut également se produire si la réponse SAML n'inclut pas de `Subject` contenant un `NameID`.

Pour plus d'informations, consultez Permettre aux principaux fédérés SAML 2.0 d'accéder au Console de gestion AWS et Configuration d'assertions SAML pour la réponse d'authentification. Pour afficher la réponse SAML dans votre navigateur, suivez les étapes répertoriées dans la section Affichage d'une réponse SAML dans votre navigateur.

Erreur : RoleSessionName le champ AuthnResponse doit correspondre à [a-z A-Z _0-9+=, .@-] {2,64} (service : AWSSecurityTokenService ; code d'état : 400 ; code d'erreur :) InvalidIdentityToken

Cette erreur peut se produire si la valeur d'attribut `RoleSessionName` est trop longue ou contient des caractères non valides. La longueur valide maximale est de 64 caractères.

Pour plus d'informations, veuillez consulter Configuration d'assertions SAML pour la réponse d'authentification. Pour afficher la réponse SAML dans votre navigateur, suivez les étapes répertoriées dans la section Affichage d'une réponse SAML dans votre navigateur.

Erreur : l'identité de la source doit correspondre à [a-z A-Z _0-9+=, .@-] {2,256} et ne pas commencer par **« `aws :` »** (service : ; code d'état : 400 AWSSecurityTokenService ; code d'erreur :) InvalidIdentityToken

Cette erreur peut se produire si la valeur d'attribut `sourceIdentity` est trop longue ou contient des caractères non valides. La longueur maximale valide est de 256 caractères. Pour de plus amples informations sur l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.

Pour de plus amples informations sur la création d'assertions SAML, veuillez consulter Configuration d'assertions SAML pour la réponse d'authentification. Pour afficher la réponse SAML dans votre navigateur, suivez les étapes répertoriées dans la section Affichage d'une réponse SAML dans votre navigateur.

Erreur : signature de réponse non valide (service : AWSSecurityTokenService ; code d'état : 400 ; code d'erreur : InvalidIdentityToken)

Cette erreur peut se produire lorsque les métadonnées du fournisseur d'identité ne correspondent pas aux métadonnées du fournisseur d'identité IAM. Par exemple, le fichier de métadonnées du fournisseur de service d'identité peut avoir changé pour mettre à jour un certificat arrivé à expiration. Téléchargez le fichier de métadonnées SAML mis à jour depuis votre fournisseur de service d'identité. Mettez-le ensuite à jour dans l'entité du fournisseur AWS d'identité que vous définissez dans IAM à l'aide de la commande CLI `aws iam update-saml-provider` multiplateforme ou de l'`Update-IAMSAMLProvider` PowerShellapplet de commande.

Erreur : clé privée non valide.

Cette erreur peut se produire si vous ne formatez pas correctement votre fichier de clé privée. Cette erreur peut fournir des informations supplémentaires sur les raisons pour lesquelles la clé privée n'est pas valide :

- La clé est chiffrée.
- Le format de la clé n'est pas reconnu. Le fichier de clé privée doit être un fichier .pem.

Lorsque vous êtes Création d'un fournisseur d'identité SAML dans IAM dans le Console de gestion AWS, vous devez télécharger la clé privée auprès de votre fournisseur d'identité pour la fournir à IAM afin d'activer le chiffrement. La clé privée doit être un fichier .pem qui utilise AES-GCM un algorithme de AES-CBC chiffrement pour déchiffrer les assertions SAML.

Erreur : échec de la suppression de la clé privée.

Cette erreur peut se produire lorsque le chiffrement SAML est défini sur Obligatoire et que votre requête supprimerait la seule clé de déchiffrement privée pour le fournisseur SAML IAM. Pour plus d'informations sur la rotation des clés privées, consultez Gestion des clés de chiffrement SAML.

Erreur : impossible de supprimer la clé privée, car l'ID de la clé ne correspond pas à une clé privée.

Cette erreur peut se produire si la valeur `keyId` de la clé privée ne correspond à aucun des ID de clé des fichiers de clé privée du fournisseur d'identité.

Lorsque vous utilisez les opérations d'API update-saml-provider ou UpdateSAMLProvider pour supprimer des clés privées de chiffrement SAML, la valeur `RemovePrivateKey` doit être un ID de clé valide pour une clé privée attachée à votre fournisseur d'identité.

Erreur : échec de l'attribution du rôle : émetteur absent du fournisseur spécifié (service : AWSOpenIdDiscoveryService ; code d'état : 400 ; code d'erreur : AuthSamlInvalidSamlResponseException)

Cette erreur peut se produire si l'émetteur de la réponse SAML ne correspond pas à l'émetteur déclaré dans le fichier de métadonnées de fédération. Le fichier de métadonnées a été chargé AWS lorsque vous avez créé le fournisseur d'identité dans IAM.

Erreur : Impossible d'analyser les métadonnées.

Cette erreur peut se produire si vous ne formatez pas correctement votre fichier de métadonnées.

Lorsque vous créez ou gérez un fournisseur d'identité SAML dans le Console de gestion AWS, vous devez récupérer le document de métadonnées SAML auprès de votre fournisseur d'identité.

Ce fichier de métadonnées inclut le nom de l'auteur, des informations d'expiration et des clés qui peuvent être utilisées pour valider les réponses d'authentification (assertions) SAML reçues du fournisseur d'identité (IdP). Le fichier de métadonnées doit être codé dans un UTF-8 format sans marque d'ordre d'octet (BOM). Pour supprimer la nomenclature, vous pouvez encoder le fichier à l' UTF-8 aide d'un outil d'édition de texte, tel que Notepad++.

Le X.509 certificat inclus dans le document de métadonnées SAML doit utiliser une taille de clé d'au moins 1024 bits. En outre, le X.509 certificat doit également être exempt de toute extension répétée. Vous pouvez utiliser des extensions, mais elles ne peuvent apparaître qu'une seule fois dans le certificat. Si le X.509 certificat ne remplit aucune des conditions, la création de l'IdP échoue et renvoie une erreur « Impossible d'analyser les métadonnées ».

Tel que défini par le profil d'interopérabilité V2.0 des métadonnées SAML version 1.0, IAM n'évalue ni ne prend aucune mesure en cas d'expiration des X.509 certificats contenus dans les documents de métadonnées SAML. Si vous êtes préoccupé par l'expiration X.509 des certificats, nous vous recommandons de surveiller les dates d'expiration des certificats et de procéder à une rotation des certificats conformément aux politiques de gouvernance et de sécurité de votre organisation.

Erreur : impossible de mettre à jour le fournisseur d'identité. Aucune mise à jour n'est définie pour les métadonnées ou l'assertion de chiffrement.

Cette erreur peut se produire si vous utilisez les opérations de l'`update-saml-provider` CLI ou de l'API `UpdateSAMLProvider`, mais que vous ne fournissez pas de valeurs de mise à jour dans les paramètres de votre requête. Pour plus d'informations sur la mise à jour de votre fournisseur SAML IAM, consultez. Création d'un fournisseur d'identité SAML dans IAM

Erreur : impossible de définir le mode de chiffrement des assertions sur Obligatoire, car aucune clé privée n'est fournie.

Cette erreur peut se produire lorsque vous n'avez pas encore chargé de clé de déchiffrement privée et que vous tentez de définir le chiffrement SAML sur Obligatoire sans inclure de clé privée dans votre requête.

Assurez-vous qu'une clé privée est définie pour votre fournisseur SAML IAM lorsque vous utilisez une `create-saml-provider` CLI, une API `CreateSAMLProvider`, une `update-saml-provider` CLI ou des opérations d'API `UpdateSAMLProvider` pour exiger des assertions SAML chiffrées.

Erreur : impossible d'ajouter et de supprimer des clés privées dans la même requête. Définissez une valeur pour un seul des deux paramètres.

Cette erreur peut se produire si des valeurs d'ajout et de suppression de clés privées sont incluses dans la même requête.

Lorsque vous utilisez les opérations API update-saml-provider ou UpdateSAMLProvider pour effectuer la rotation des fichiers de clés privées de chiffrement SAML, vous ne pouvez qu'ajouter ou supprimer une clé privée dans votre requête. Si vous ajoutez une clé privée tout en supprimant une clé privée, l'opération échoue. Pour plus d'informations sur la rotation des clés privées, consultez Gestion des clés de chiffrement SAML.

Erreur : Le fournisseur spécifié n'existe pas.

Cette erreur peut se produire si le nom du fournisseur dans l'assertion SAML ne correspond pas au nom du fournisseur dans IAM. Pour plus d'informations sur l'affichage du nom du fournisseur, veuillez consulter Création d'un fournisseur d'identité SAML dans IAM.

Erreur : la DurationSeconds valeur demandée dépasse la valeur MaxSessionDuration définie pour ce rôle.

Cette erreur peut se produire si vous assumez un rôle depuis l'API AWS CLI or.

Lorsque vous utilisez l'interface de ligne de commande ou les opérations d'AssumeRoleWithSAMLAPI assume-role-with-saml pour assumer un rôle, vous pouvez spécifier une valeur pour le paramètre. `DurationSeconds` Vous pouvez spécifier une valeur comprise entre 900 secondes (15 minutes) et la durée maximale de la session définie pour le rôle. Si vous spécifiez une valeur supérieure à ce paramètre, l'opération échoue. Par exemple, si vous spécifiez une durée de session de 12 heures, mais que votre administrateur a défini la durée de session maximale à 6 heures, votre opération échoue. Pour savoir comment afficher la valeur maximale pour votre rôle, veuillez consulter Mettre à jour la durée de session maximale pour un rôle.

Erreur : la limite de deux clés privées est atteinte.

Cette erreur peut se produire si vous essayez d'ajouter une clé privée à votre fournisseur d'identité.

Vous pouvez enregistrer jusqu’à deux clés privées pour chaque fournisseur d’identité. Lorsque vous utilisez les opérations d’API update-saml-provider ou UpdateSAMLProvider pour ajouter une troisième clé privée, l’opération échoue.

Supprimez les clés privées expirées avant d’ajouter une nouvelle clé privée. Pour plus d’informations sur la rotation des clés privées, consultez Gestion des clés de chiffrement SAML.

Erreur : la réponse ne contient pas l'audience requise.

Cette erreur peut se produire s'il existe une incompatibilité entre l'URL d'audience et le fournisseur d'identité dans la configuration SAML. Assurez-vous que l'identifiant de la partie dépendante de votre fournisseur d'identité (IdP) correspond exactement à l'URL d'audience (ID d'entité) fournie dans la configuration SAML.

Comment IAM travaille avec les autres services AWS

IAM est le principal AWS service que vous utiliserez pour gérer les ressources IAM, mais tous les autres AWS services fonctionnent avec IAM pour contrôler l'accès aux ressources de votre compte.

- CloudFormation

  CloudFormation s'intègre à IAM en vous permettant de définir et de gérer les ressources IAM dans le cadre de vos CloudFormation modèles. Vous pouvez l'utiliser CloudFormation pour spécifier les autorisations IAM nécessaires pour les autres AWS ressources que vous mettez en service. CloudFormation prend également en charge l'utilisation de rôles IAM pour gérer les informations d'identification requises pour le provisionnement et la gestion de votre AWS infrastructure, et sa fonction de détection de dérive vous aide à préserver l'intégrité de vos configurations IAM.

- AWS CloudShell

  Lorsque vous accédez AWS CloudShell, votre authentification et votre autorisation sont gérées par IAM. AWS CloudShell s'exécute dans le contexte d'un rôle IAM attribué à votre utilisateur ou à votre compte. Lorsque vous le lancez AWS CloudShell, il génère automatiquement des informations d'identification de sécurité temporaires en fonction du rôle IAM qui vous est attribué.

- AWS SDKs

  AWS SDKs Travaillez avec IAM en gérant le processus d'authentification et d'autorisation, en gérant les AWS informations d'identification et en respectant les autorisations et les politiques définies dans IAM pour garantir que votre application ne peut accéder qu'aux ressources qu'elle est autorisée à utiliser. Ils SDKs fournissent des mécanismes permettant d'obtenir et d'utiliser des informations d'identification de sécurité temporaires, ainsi que de valider les autorisations requises pour le fonctionnement de votre application.

Pour obtenir la liste des AWS services compatibles avec IAM et des fonctionnalités IAM prises en charge, voir. AWS services qui fonctionnent avec IAM

Créez des ressources IAM avec CloudFormation

Gestion des identités et des accès AWS est intégré à AWS CloudFormation un service qui vous aide à modéliser et à configurer vos AWS ressources afin que vous puissiez passer moins de temps à créer et à gérer vos ressources et votre infrastructure. Vous créez un modèle qui décrit toutes les

AWS ressources souhaitées (telles que les clés d'accès, les groupes, les politiques de groupe, les profils d'instance, les politiques gérées, les fournisseurs OIDC, les politiques intégrées, les rôles, les politiques de rôle, les fournisseurs SAML, les certificats de serveur, les rôles liés aux services, les utilisateurs (et l'ajout d'utilisateurs aux groupes), les politiques utilisateur et les dispositifs MFA virtuels), puis vous CloudFormation approvisionnez et configurez ces ressources pour vous.

Lorsque vous l'utilisez CloudFormation, vous pouvez réutiliser votre modèle pour configurer vos ressources IAM de manière cohérente et répétée. Décrivez vos ressources une seule fois, puis distribuez les mêmes ressources encore et encore dans plusieurs Comptes AWS régions.

IAM et modèles CloudFormation

Pour allouer et configurer des ressources pour IAM et les services associés, vous devez maîtriser les modèles CloudFormation. Les modèles sont des fichiers texte formatés en JSON ou YAML. Ces modèles décrivent les ressources que vous souhaitez mettre à disposition dans vos CloudFormation piles. Si vous n'êtes pas familiarisé avec JSON ou YAML, vous pouvez utiliser CloudFormation Designer pour vous aider à démarrer avec les CloudFormation modèles. Pour plus d'informations, consultez Qu'est-ce que CloudFormation Designer dans le Guide de l'utilisateur AWS CloudFormation .

IAM prend en charge la création de clés d'accès, de groupes, de politiques de groupe, de profils d'instance, de politiques gérées, de fournisseurs OIDC, de politiques intégrées, de rôles, de politiques de rôle, de fournisseurs SAML, de certificats de serveur, de rôles liés à des services, d'utilisateurs (et d'ajout d'utilisateurs à des groupes), de politiques utilisateur et de dispositifs MFA virtuels. CloudFormation Pour plus d'informations, notamment des exemples de modèles JSON et YAML pour vos ressources IAM, consultez Référence du type de ressource Gestion des identités et des accès AWS dans le Guide de l'utilisateur AWS CloudFormation .

Vous pouvez également créer des modèles qui créent des ressources connexes, telles que des rôles et des politiques gérées.

En savoir plus sur CloudFormation

Pour en savoir plus CloudFormation, consultez les ressources suivantes :

- AWS CloudFormation
- AWS CloudFormation Guide de l'utilisateur
- CloudFormation API Reference

- AWS CloudFormation Guide de l'utilisateur de l'interface de ligne de commande

Utilisation AWS CloudShell pour travailler avec AWS Gestion de l’identité et des accès

AWS CloudShell est un shell pré-authentifié basé sur un navigateur que vous pouvez lancer directement depuis. Console de gestion AWS Vous pouvez exécuter AWS CLI des commandes sur des AWS services (y compris la gestion des AWS identités et des accès) à l'aide de votre shell préféré (Bash PowerShell ou Z shell). Et vous pouvez le faire sans télécharger ou installer des outils de ligne de commande.

Vous lancez AWS CloudShell à partir du Console de gestion AWS, et les AWS informations d'identification que vous avez utilisées pour vous connecter à la console sont automatiquement disponibles dans une nouvelle session shell. Cette pré-authentification des AWS CloudShell utilisateurs vous permet de ne pas configurer les informations d'identification lorsque vous interagissez avec AWS des services tels qu'IAM utilisant AWS CLI la version 2 (préinstallée dans l'environnement informatique du shell).

Obtenir les autorisations IAM pour AWS CloudShell

À l'aide des ressources de gestion d'accès fournies par Gestion des identités et des accès AWS, les administrateurs peuvent accorder des autorisations aux utilisateurs IAM afin qu'ils puissent accéder aux fonctionnalités de l'environnement AWS CloudShell et les utiliser.

Le moyen le plus rapide pour un administrateur d'accorder l'accès aux utilisateurs est d'utiliser une politique AWS gérée. Une politique gérée par AWS est une politique autonome qui est créée et gérée par AWS. La politique AWS gérée suivante CloudShell peut être associée aux identités IAM :

- `AWSCloudShellFullAccess`: autorise l'utilisation AWS CloudShell avec un accès complet à toutes les fonctionnalités.

Si vous souhaitez limiter l'étendue des actions qu'un utilisateur IAM peut effectuer AWS CloudShell, vous pouvez créer une stratégie personnalisée qui utilise la stratégie `AWSCloudShellFullAccess` gérée comme modèle. Pour plus d'informations sur la limitation des actions disponibles pour les utilisateurs CloudShell, consultez la section Gestion de l' AWS CloudShell accès et de l'utilisation à l'aide des politiques IAM dans le Guide de l'AWS CloudShell utilisateur.

Interaction avec IAM

Après le lancement AWS CloudShell depuis le Console de gestion AWS, vous pouvez immédiatement commencer à interagir avec IAM à l'aide de l'interface de ligne de commande.

> **Note**
>
> Lorsque vous AWS CLI l'utilisez AWS CloudShell, vous n'avez pas besoin de télécharger ou d'installer de ressources supplémentaires. De plus, comme vous êtes déjà authentifié dans le shell, vous n'avez pas besoin de configurer les informations d'identification avant d'effectuer des appels.

Créez un groupe IAM et ajoutez un utilisateur IAM au groupe à l'aide de AWS CloudShell

L'exemple suivant permet CloudShell de créer un groupe IAM, d'ajouter un utilisateur IAM au groupe, puis de vérifier que la commande a réussi.

1. À partir de Console de gestion AWS, vous pouvez lancer CloudShell en choisissant les options suivantes disponibles dans la barre de navigation :
   - Choisissez l' CloudShell icône.
   - Commencez à taper « cloudshell » dans la zone de recherche, puis choisissez l' CloudShelloption.
2. Pour créer un groupe IAM, entrez la commande suivante sur la ligne de CloudShell commande. Dans cet exemple, nous avons nommé le groupe east_coast :

   ```
   aws iam create-group --group-name east_coast
   ```

   Si l'appel aboutit, la ligne de commande affiche une réponse du service similaire au résultat suivant :

   ```
   {
      "Group": {
          "Path": "/",
          "GroupName": "east_coast",
          "GroupId": "AGPAYBDBW4JBY3EXAMPLE",
          "Arn": "arn:aws:iam::111122223333:group/east_coast",
   ```

```
            "CreateDate": "2023-09-11T21:02:21+00:00"
        }
    }
```

3. Pour ajouter un utilisateur au groupe que vous avez créé, utilisez la commande suivante, en spécifiant le nom du groupe et le nom d'utilisateur. Dans cet exemple, nous avons nommé le groupe east_coast et l'utilisateur john :

```
aws iam add-user-to-group --group-name east_coast --user-name john
```

4. Pour vérifier que l'utilisateur fait partie du groupe, utilisez la commande suivante en spécifiant le nom du groupe. Dans cet exemple, nous continuons à utiliser le groupe east_coast :

```
aws iam get-group --group-name east_coast
```

Si l'appel aboutit, la ligne de commande affiche une réponse du service similaire au résultat suivant :

```
{
  "Users": [
    {
        "Path": "/",
        "UserName": "john",
        "UserId": "AIDAYBDBW4JBXGEXAMPLE",
        "Arn": "arn:aws:iam::552108220995:user/john",
        "CreateDate": "2023-09-11T20:43:14+00:00",
        "PasswordLastUsed": "2023-09-11T20:59:14+00:00"
    }
  ],
  "Group": {
        "Path": "/",
        "GroupName": "east_coast",
        "GroupId": "AGPAYBDBW4JBY3EXAMPLE",
        "Arn": "arn:aws:iam::111122223333:group/east_coast",
        "CreateDate": "2023-09-11T21:02:21+00:00"
    }
}
```

L'utilisation de ce service avec un AWS Kit SDK

AWS des kits de développement logiciel (SDK) sont disponibles pour de nombreux langages de programmation populaires. Chaque kit SDK fournit une API, des exemples de code et de la documentation qui facilitent la création d'applications par les développeurs dans leur langage préféré.

Documentation SDK	Exemples de code
AWS SDK pour C++	AWS SDK pour C++ exemples de code
AWS CLI	AWS CLI exemples de code
AWS SDK pour Go	AWS SDK pour Go exemples de code
AWS SDK pour Java	AWS SDK pour Java exemples de code
AWS SDK pour JavaScript	AWS SDK pour JavaScript exemples de code
AWS SDK pour Kotlin	AWS SDK pour Kotlin exemples de code
AWS SDK pour .NET	AWS SDK pour .NET exemples de code
AWS SDK pour PHP	AWS SDK pour PHP exemples de code
Outils AWS pour PowerShell	Outils AWS pour PowerShell exemples de code
AWS SDK pour Python v4	AWS Exemples de code SDK pour Python v4
AWS SDK pour Python (Boto3)	AWS SDK pour Python (Boto3) exemples de code
AWS SDK pour Ruby	AWS SDK pour Ruby exemples de code
AWS SDK pour Rust	AWS SDK pour Rust exemples de code
AWS SDK pour SAP ABAP	AWS SDK pour SAP ABAP exemples de code
AWS SDK pour Swift	AWS SDK pour Swift exemples de code

Pour voir des exemples spécifiques à ce service, consultez Exemples de code pour l'utilisation d'IAM AWS Kits SDK.

Exemple de disponibilité

Vous n’avez pas trouvé ce dont vous avez besoin ? Demandez un exemple de code en utilisant le lien Provide feedback (Fournir un commentaire) en bas de cette page.

Informations de référence pour Gestion des identités et des accès AWS

Utilisez les rubriques de cette section pour trouver des documents de référence détaillés sur différents aspects d'IAM et AWS STS.

Rubriques

- Identifier AWS ressources avec Amazon Resource Names (ARN)
- Identifiants IAM
- IAM et AWS STS quotas
- Prise en charge des points de terminaison double pile
- Points de terminaison de VPC d'Interface
- AWS services qui fonctionnent avec IAM
- AWS Signature Version 4 pour les demandes d'API
- Référence de politique JSON IAM

- Informations d' Service AWS simplifiées pour un accès programmatique

Identifier AWS ressources avec Amazon Resource Names (ARN)

Les Amazon Resource Names (ARN) identifient les AWS ressources de manière unique. Nous avons besoin d'un ARN lorsque vous devez spécifier une ressource sans ambiguïté dans tous les domaines AWS, par exemple dans les politiques IAM, les balises Amazon Relational Database Service (Amazon RDS) et les appels d'API. Bien que les ARN, comme toute information d'identification, doivent être utilisés et partagés avec prudence, ils ne sont pas considérés comme des informations secrètes, sensibles ou confidentielles.

Format ARN

Voici les formats généraux des ARN. Les formats spécifiques dépendent de la ressource. Pour utiliser un ARN, remplacez le *`italicized`* texte par les informations spécifiques à la ressource. Sachez que les ARN de certaines ressources omettent la région, l'ID du compte ou la région et l'ID du compte.

```
arn:partition:service:region:account-id:resource-id
arn:partition:service:region:account-id:resource-type/resource-id
arn:partition:service:region:account-id:resource-type:resource-id
```

partition

Partition dans laquelle se trouve la ressource. Une partition est un groupe de AWS régions. Chaque AWS compte est limité à une partition.

Les partitions prises en charge sont les suivantes :

- `aws`- AWS Régions
- `aws-cn` - Régions chinoises
- `aws-us-gov` - AWS GovCloud (US) Régions

service

L'espace de noms du service qui identifie le AWS produit.

region

Code de région. Par exemple, indiquez `us-east-2` pour la région USA Est (Ohio). Pour obtenir la liste des codes de région, consultez Points de terminaison régionaux dans Références générales AWS.

account-id

L'ID du AWS compte propriétaire de la ressource, sans les tirets. Par exemple, `123456789012`.

resource-type

Type de ressource. Par exemple, `vpc` pour un cloud privé virtuel (VPC)

resource-id

Identificateur de la ressource. Il s'agit du nom de la ressource, de l'ID de la ressource ou d'un chemin de ressource. Certains identificateurs de ressources incluent une ressource parente (sous-ressource- type/parent - resource/sub -ressource) ou un qualificatif tel qu'une version (resource-type:resource-name:qualifier).

Exemples

Utilisateur IAM

arn:aws:iam : ::user/ *123456789012 john*

Rubrique SNS

arn:aws:sns : : *us-east-1 123456789012 example-sns-topic-name*

VPC

arn:aws:ec2 : ::vpc/ *us-east-1 123456789012 vpc-0e9801d129EXAMPLE*

Rechercher le format d'ARN pour une ressource

Le format exact d'un ARN dépend du service et du type de ressource. Certains ARN de ressources peuvent inclure un chemin, une variable ou un caractère générique. Pour rechercher le format ARN d'une AWS ressource spécifique, ouvrez la référence d'autorisation de service, ouvrez la page du service et accédez au tableau des types de ressources.

Chemins d'accès dans les ARN

Certains ARN de ressource peuvent inclure un chemin. Par exemple, dans Amazon S3, l'identificateur de ressource est un nom d'objet qui peut inclure des barres obliques (`/`) pour former un chemin d'accès. De même, les noms d'utilisateur et les noms de groupe IAM peuvent inclure des chemins d'accès. Seuls les caractères alphanumériques et les caractères suivants sont autorisés dans les chemins IAM : barre oblique (`/`), plus (`+`), égal (`=`), virgule (`,`), point (`.`), arobase (`@`), trait de soulignement (`_`), et trait d'union (`-`).

Utilisation de caractères génériques dans les chemins

Les chemins peuvent inclure un caractère générique, à savoir un astérisque (`*`). Certains éléments de politique autorisent les caractères génériques, d'autres non. Vous pouvez utiliser des caractères génériques pour la ressource ou les NotResource éléments, mais pas pour le principal ou les NotPrincipal éléments. Pour de plus amples informations, veuillez consulter Référence de politique JSON IAM.

Vous pouvez spécifier `role/*` pour désigner tous les rôles du compte 123456789012, comme dans l'exemple suivant :

```
arn:aws:iam::123456789012:role/*
```

Vous pouvez également terminer le nom d'une ressource par un caractère générique. Par exemple, vous pouvez spécifier **`service-*`** pour désigner tous les rôles commençant par `service` et se terminant par différents caractères tels que `service-role1` ou `service-test` :

```
arn:aws:iam::123456789012:role/service-*
```

L'exemple suivant illustre les ARN pour les objets d'un compartiment Amazon S3 dans lequel le nom de la ressource inclut un chemin d'accès. L'ARN `arn:aws:s3:::amzn-s3-demo-bucket/*` concerne tous les objets de ce compartiment, quel que soit leur préfixe. L'ARN `arn:aws:s3:::amzn-s3-demo-bucket/Development/*` concerne tous les objets créés dans le préfixe **`/Development/`**.

Vous pouvez également utiliser le caractère générique ? pour spécifier un caractère dans un ARN. Par exemple, vous pouvez utiliser l'ARN suivant pour tous les dossiers commençant par quatre caractères et se terminant par **`-test`** dans le compartiment S3 nommé amzn-s3-demo-bucket. Certains dossiers qui correspondraient à cette description incluent `1234-test`, `2024-test`, ou `a100-test`.

```
arn:aws:s3:::amzn-s3-demo-bucket/????-test
```

Vous pouvez également utiliser des caractères génériques dans les différentes sections d'un ARN, délimitées par deux points « ». **`:`** Dans l'exemple suivant, deux caractères génériques sont utilisés pour correspondre à toutes les applications Amazon Q et ressources au sein des applications dans toutes les régions pour le compte 123456789012 :

```
arn:aws:qbusiness:*:123456789012:*
```

De même, l'exemple suivant correspond à tous les VPC Amazon dans toutes les régions pour le compte 123456789012 :

```
arn:aws:ec2:*:123456789012:vpc/*
```

L'exemple suivant correspond à tous les volumes Amazon EBS dans toutes les régions pour le compte 123456789012 :

```
arn:aws:ec2:*:123456789012:volume/*
```

Limitations relatives à l'utilisation des caractères génériques dans les ARN

Vous ne pouvez pas utiliser de caractère générique dans la partie de l'ARN qui spécifie le type de ressource. L'exemple d'ARN suivant, qui contient un caractère générique dans le type de ressource, n'est pas valide :

```
arn:aws:lambda:us-east-2:123456789012:functi*:my-function <== not allowed
```

> **Note**
>
> Lorsque vous spécifiez un ARN incomplet (un contenant moins de six champs standard) dans une politique basée sur l'identité, complète AWS automatiquement l'ARN en ajoutant des caractères génériques (*) à tous les champs manquants. Par exemple, spécifier `arn:aws:sqs` est équivalent à`arn:aws:sqs:*:*:*`, ce qui donne accès à toutes les ressources Amazon Amazon SQS dans toutes les régions et tous les comptes. Cependant, les politiques de session transmises et AWS STS AssumeRole AssumeRoleWithWebIdentity les AssumeRoleWithSAML demandes ne prennent pas en charge les ARN incomplets. L'utilisation d'un ARN incomplet dans une politique de session entraînera une `MalformedPolicyDocumentException` erreur.

Vous ne pouvez pas non plus utiliser de caractère générique dans le préfixe de l'ARN, ni avoir de caractère générique dans la section partition d'un ARN.

```
arn:aws:redshift:us-east-1:123456789012:? <== not allowed
```

Identifiants IAM

IAM utilise quelques identifiants différents pour les utilisateurs, les groupes IAM, les rôles, les politiques et les certificats de serveur. Cette section décrit les identifiants et quand vous utilisez chacun d'eux.

Rubriques

- Noms conviviaux et chemins

- ARN IAM
- Identifiants uniques
- Présentation des préfixes d'ID uniques
- Obtention de l'identifiant unique

Noms conviviaux et chemins

Lorsque vous créez un utilisateur, un rôle, un groupe d'utilisateurs ou une politique, ou lorsque vous téléchargez un certificat de serveur, vous lui attribuez un nom convivial. Les exemples incluent Bob, TestApp 1 an, Developers ManageCredentialsPermissions, ou ProdServerCert.

Si vous utilisez l'API IAM ou AWS Command Line Interface (AWS CLI) pour créer des ressources IAM, vous pouvez ajouter un chemin facultatif. Vous pouvez utiliser un seul chemin ou imbriquer plusieurs chemins en une structure de dossiers. Par exemple, vous pouvez utiliser le chemin imbriqué `/division_abc/subdivision_xyz/product_1234/engineering/` pour correspondre à l'organigramme de votre société. Vous pouvez ensuite créer une politique pour autoriser tous les utilisateur dans ce chemin à accéder à l'API de simulateur de politique. Pour afficher cette politique, consultez IAM : accès à l'API du simulateur de politique en fonction du chemin d'utilisateur. Pour plus d'informations sur la façon dont un nom convivial peut être spécifié, consultez la documentation de l'API utilisateur. Pour trouver des exemples de la manière dont vous pouvez utiliser des chemins, consultez ARN IAM.

Lorsque vous créez CloudFormation des ressources, vous pouvez spécifier un chemin pour les utilisateurs, les groupes et rôles IAM, ainsi que les politiques gérées par le client.

Si vous avez un utilisateur et un groupe d'utilisateurs sur le même chemin, IAM ne place pas automatiquement l'utilisateur dans ce groupe d'utilisateurs. Par exemple, vous pouvez créer un groupe d'utilisateurs Développeurs et spécifier le chemin d'accès comme `/division_abc/subdivision_xyz/product_1234/engineering/`. Si vous créez un utilisateur nommé Bob et lui ajoutez le même chemin, cela ne place pas Bob automatiquement dans le groupe d'utilisateurs Développeurs. IAM n'impose aucune limite entre utilisateurs ou groupes IAM en fonction de leurs chemins. Les utilisateurs ayant des chemins différents peuvent utiliser les mêmes ressources en supposant qu'ils aient autorisation d'accéder à ces ressources. Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour de plus amples informations, veuillez consulter IAM et AWS STS quotas.

ARN IAM

Les plupart des ressources ont un nom convivial, par exemple, un utilisateur appelé `Bob` ou un groupe d'utilisateurs appelé `Developers`. Toutefois, le langage de politique d'autorisation nécessite que vous indiquiez la ou les ressources au format Amazon Resource Name (ARN) suivant.

```
arn:partition:service:region:account:resource
```

Où :

- `partition` identifie la partition pour la ressource. Pour les régions AWS standard, la partition est `aws`. Si vous avez des ressources dans d'autres partitions, la partition est `aws-`*`partitionname`*. Par exemple, la partition des ressources de la région Chine (Beijing) est `aws-cn`. Vous ne pouvez pas déléguer l'accès entre comptes dans différentes partitions.
- `service`identifie le AWS produit. Les ressources IAM utilisent toujours `iam`.
- `region` identifie la région de la ressource. Pour les ressources IAM, elle est toujours laissée vide.
- `account`spécifie l' Compte AWS identifiant sans tiret.
- `resource` identifie la ressources spécifique par nom.

Vous pouvez spécifier IAM et ARN à l' AWS STS aide de la syntaxe suivante. La partie de la région de l'ARN est vide, car les ressources IAM sont globales.

Syntaxe :

```
arn:aws:iam::account:root
arn:aws:iam::account:user/user-name-with-path
arn:aws:iam::account:group/group-name-with-path
arn:aws:iam::account:role/role-name-with-path
arn:aws:iam::account:policy/policy-name-with-path
arn:aws:iam::account:instance-profile/instance-profile-name-with-path
arn:aws:sts::account:federated-user/user-name
arn:aws:sts::account:assumed-role/role-name/role-session-name
arn:aws:sts::account:self
arn:aws:iam::account:mfa/virtual-device-name-with-path
arn:aws:iam::account:u2f/u2f-token-id
arn:aws:iam::account:server-certificate/certificate-name-with-path
arn:aws:iam::account:saml-provider/provider-name
arn:aws:iam::account:oidc-provider/provider-name
```

```
arn:aws:iam::aws:contextProvider/context-provider-name
```

Bon nombre des exemples suivants incluent des chemins dans la partie des ressources de l'ARN. Les chemins ne peuvent pas être créés ou manipulés dans l'interface Console de gestion AWS. Pour utiliser les chemins, vous devez utiliser la ressource à l'aide de l' AWS API, des outils ou des outils pour Windows PowerShell. AWS CLI

Exemples :

```
arn:aws:iam::123456789012:root
arn:aws:iam::123456789012:user/John
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/Jane
arn:aws:iam::123456789012:group/Developers
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
arn:aws:iam::123456789012:role/S3Access
arn:aws:iam::123456789012:role/application_abc/component_xyz/RDSAccess
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
arn:aws:iam::123456789012:role/service-role/QuickSightAction
arn:aws:iam::123456789012:policy/UsersManageOwnCredentials
arn:aws:iam::123456789012:policy/division_abc/subdivision_xyz/UsersManageOwnCredentials
arn:aws:iam::123456789012:instance-profile/Webserver
arn:aws:sts::123456789012:federated-user/John
arn:aws:sts::123456789012:assumed-role/Accounting-Role/Jane
arn:aws:sts::123456789012:self
arn:aws:iam::123456789012:mfa/JaneMFA
arn:aws:iam::123456789012:u2f/user/John/default (U2F security key)
arn:aws:iam::123456789012:server-certificate/ProdServerCert
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
arn:aws:iam::123456789012:oidc-provider/server.example.org
arn:aws:iam::aws:contextProvider/IdentityCenter
```

Les exemples suivants fournissent plus de détails pour vous aider à comprendre le format ARN pour différents types d'IAM et de AWS STS ressources.

- Un utilisateur IAM dans le compte :

> **Note**
>
> Chaque nom d'utilisateur IAM est unique. Le nom d'utilisateur est sensible à la casse pour l'utilisateur, par exemple lors du processus de connexion, mais il l'est également lorsque vous l'utilisez dans une politique ou dans le cadre d'un ARN.

```
arn:aws:iam::123456789012:user/John
```

- Un autre utilisateur avec un chemin reflétant un organigramme :

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/Jane
```

- Pour un groupe d'utilisateurs IAM :

```
arn:aws:iam::123456789012:group/Developers
```

- Un groupe d'utilisateurs IAM avec un chemin :

```
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
```

- Un rôle IAM :

```
arn:aws:iam::123456789012:role/S3Access
```

- Un rôle lié à un service :

```
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
```

- Un rôle de service :

```
arn:aws:iam::123456789012:role/service-role/QuickSightAction
```

- Une politique gérée :

```
arn:aws:iam::123456789012:policy/ManageCredentialsPermissions
```

- Un profil d'instance qui peut être associé à une instance Amazon EC2 :

```
arn:aws:iam::123456789012:instance-profile/Webserver
```

- Un utilisateur principal AWS STS fédéré identifié dans IAM comme « Paulo » :

```
arn:aws:sts::123456789012:federated-user/Paulo
```

- La session active d'une personne assumant le rôle de Accounting-Role « », dont le nom de session de rôle est « Mary » :

```
arn:aws:sts::123456789012:assumed-role/Accounting-Role/Mary
```

- Représente la propre session de l'appelant lorsqu'elle est utilisée comme ressource dans un appel d'API, tel que l' AWS STS SetContextAPI, qui fonctionne sur la session d'appel :

```
arn:aws:sts::123456789012:self
```

- L'appareil MFA (Multi-Factor Authentication) attribué à l'utilisateur nommé Jorge :

```
arn:aws:iam::123456789012:mfa/Jorge
```

- Un certificat de serveur :

```
arn:aws:iam::123456789012:server-certificate/ProdServerCert
```

- Un certificat de serveur avec un chemin qui reflète un organigramme :

```
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
```

- Fournisseurs d'identité (SAML et OIDC) :

```
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
arn:aws:iam::123456789012:oidc-provider/server.example.org
```

- Un fournisseur d'identité OIDC avec un chemin qui reflète l'URL d'un fournisseur d'identité OIDC d'Amazon EKS :

```
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
```

- L'ARN du fournisseur de contexte utilisé pour la propagation d'identité AWS IAM Identity Center sécurisée, à partir duquel l'assertion de contexte fiable a été générée :

```
arn:aws:iam::aws:contextProvider/IdentityCenter
```

Autre ARN important : celui de l'utilisateur racine. Bien qu'il ne s'agisse pas d'une ressource IAM, vous devez connaître le format de cet ARN. Il est souvent utilisé dans l'élément `Principal` d'une stratégie.

- Les éléments suivants Compte AWS s'affichent :

```
arn:aws:iam::123456789012:root
```

L'exemple suivant montre une politique que vous pouvez affecter à Richard afin de lui permettre de gérer ses clés d'accès. Notez que la ressource est l'utilisateur IAM Richard.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ManageRichardAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/division_abc/subdivision_xyz/
Richard"
        },
        {
            "Sid": "ListForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "*"
        }
    ]
```

```
}
```

> **Note**
>
> Lorsque vous utilisez des ARN pour identifier des ressources dans une politique IAM, vous pouvez inclure des variables de politique. Les variables de politique peuvent inclure des espaces réservés pour les informations d'exécution (notamment le nom de l'utilisateur) comme faisant partie de l'ARN. Pour plus d'informations, veuillez consulter Éléments des politiques IAM : variables et balises.

Utilisation de caractères génériques et de chemins dans les ARN

Vous pouvez utiliser des caractères génériques dans la *resource* partie de l'ARN pour spécifier plusieurs utilisateurs ou groupes ou politiques IAM. Par exemple, pour spécifier tous les utilisateurs travaillant sur le produit_1234, vous utilisez :

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/*
```

Si vous avez des utilisateurs dont le nom commence par la chaîne `app_`, vous pouvez faire référence à tous ces utilisateurs avec l'ARN suivant.

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/app_*
```

Pour spécifier tous les utilisateurs, groupes IAM ou politiques de votre Compte AWS, utilisez un caractère générique après le `user/group/`, ou une `policy/` partie de l'ARN, respectivement.

```
arn:aws:iam::123456789012:user/*
arn:aws:iam::123456789012:group/*
arn:aws:iam::123456789012:policy/*
```

Si vous spécifiez l'ARN suivant pour un utilisateur `arn:aws:iam::111122223333:user/*`, il correspond aux deux exemples suivants.

```
arn:aws:iam::111122223333:user/John
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/JaneDoe
```

Mais, si vous spécifiez l'ARN suivant pour un utilisateur `arn:aws:iam::111122223333:user/division_abc*`, il correspond au second exemple, mais pas au premier.

```
arn:aws:iam::111122223333:user/John
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Jane
```

N'utilisez pas un caractère générique dans la partie `user/`, `group/` ou `policy/` de l'ARN. Par exemple, IAM n'autorise pas ce qui suit :

```
arn:aws:iam::123456789012:u*
```

> **Note**
>
> Lorsque vous spécifiez un ARN incomplet (un contenant moins de six champs standard) dans une politique basée sur l'identité, complète AWS automatiquement l'ARN en ajoutant des caractères génériques (*) à tous les champs manquants. Par exemple, spécifier `arn:aws:sqs` est équivalent à`arn:aws:sqs:*:*:*`, ce qui donne accès à toutes les ressources Amazon Amazon SQS dans toutes les régions et tous les comptes. Cependant, les politiques de session transmises et AWS STS AssumeRole AssumeRoleWithWebIdentity les AssumeRoleWithSAML demandes ne prennent pas en charge les ARN incomplets. L'utilisation d'un ARN incomplet dans une politique de session entraînera une `MalformedPolicyDocumentException` erreur.

Example Exemple d'utilisation des chemins et des ARN d'un groupe d'utilisateurs basé sur un projet

Les chemins ne peuvent pas être créés ou manipulés dans l'interface Console de gestion AWS. Pour utiliser les chemins, vous devez utiliser la ressource à l'aide de l' AWS API AWS CLI, de ou des Outils pour Windows PowerShell.

Dans cet exemple, Jules du groupe Marketing_Admin d'utilisateurs crée un groupe d'utilisateurs basé sur un projet dans le chemin /marketing/. Il attribue des utilisateurs de différentes parties de l'entreprise au groupe d'utilisateurs. Cet exemple illustre que le chemin d'un utilisateur n'est pas lié aux groupes d'utilisateurs dont fait partie l'utilisateur.

Le groupe marketing a un nouveau produit qu'il va lancer. Jules crée donc un nouveau groupe d'utilisateurs dans le chemin /marketing/ appelé. Widget_Launch Jules affecte ensuite la politique suivante au groupe d'utilisateurs qui accorde à celui-ci l'accès aux objets situés dans la partie du `example_bucket` désigné pour ce lancement spécifique.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::example_bucket/marketing/newproductlaunch/widget/
*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket*",
      "Resource": "arn:aws:s3:::example_bucket",
      "Condition": {"StringLike": {"s3:prefix": "marketing/newproductlaunch/
widget/*"}}
    }
  ]
}
```

Jules affecte ensuite les utilisateurs qui travaillent sur ce lancement au groupe d'utilisateurs. Cela inclut Patricia et Eli du chemin /marketing/. Cela inclut également Chris et Chloe du chemin /sales/ et Alice et Jim du chemin /legal/.

Identifiants uniques

Lorsqu'IAM crée un utilisateur, un groupe d'utilisateurs, un rôle, une stratégie, un profil d'instance ou un certificat de serveur, il accorde à chaque ressource un ID unique. L'ID se présente comme suit :

`AIDAJQABLZS4A3QDU576Q`

En règle générale, vous avez recours à des noms conviviaux et à des ARN lorsque vous utilisez des ressources IAM. De la sorte, vous n'avez pas besoin de connaître l'ID unique d'une ressource spécifique. En revanche, l'ID unique peut parfois être utile quand il n'est pratique d'utiliser des noms conviviaux.

Un exemple réutilise des noms conviviaux dans votre Compte AWS. Dans votre compte, un nom convivial pour un utilisateur, un groupe d'utilisateurs ou une stratégie doivent être uniques. Par exemple, vous pouvez créer un utilisateur IAM appelé `John`. Votre entreprise utilise Amazon S3

et dispose d'un compartiment avec des dossiers pour chaque employé. L'utilisateur IAM `John` est membre d'un groupe d'utilisateurs IAM appelé `User-S3-Access` avec des autorisations qui permettent aux utilisateurs d'accéder uniquement à leurs propres dossiers dans le compartiment. Pour obtenir un exemple de création d'une stratégie basée sur l'identité qui autorise des utilisateurs IAM à accéder à leur propre objet de compartiment dans S3 à l'aide du nom convivial des utilisateurs, consultez Amazon S3 : autorise les utilisateurs IAM à accéder à leur répertoire de base S3, par programmation et dans la console.

Supposons que l'employé appelé John quitte votre entreprise et que vous supprimez l'utilisateur IAM correspondant appelé `John`. Mais plus tard, un autre employé appelé John commence à travailler pour votre entreprise et vous créez un utilisateur IAM appelé `John`. Vous ajoutez le nouvel utilisateur IAM appelé `John` au groupe d'utilisateurs IAM `User-S3-Access` existant. Si la stratégie associée au groupe d'utilisateurs spécifie le nom d''utilisateur IAM convivial `John`, la stratégie permet au nouveau John d'accéder aux informations laissées par l'ancien John.

En règle générale, nous vous recommandons de spécifier l'ARN de la ressource dans vos politiques au lieu de son ID unique. En revanche, chaque utilisateur IAM dispose d'un ID unique, même si vous créez un utilisateur IAM qui réutilise un nom convivial que vous avez supprimé avant. Dans l'exemple, l'ancien utilisateur IAM `John` et le nouvel utilisateur IAM `John` ont des ID uniques distincts. Vous pouvez créer des politiques basées sur les ressources qui accordent l'accès par ID unique et pas seulement par nom d'utilisateur. Vous réduisez ainsi le risque d'accorder par inadvertance l'accès à des informations dont un employé ne devrait pas disposer.

L'exemple suivant montre comment spécifier des ID uniques dans l'élément `Principal` principal d'une stratégie basée sur les ressources.

```
"Principal": {
  "AWS": [
    "arn:aws:iam::111122223333:role/role-name",
    "AIDACKCEVSQ6C2EXAMPLE",
    "AROADBQP57FF2AEXAMPLE"
  }
```

L'exemple suivant montre comment spécifier des ID uniques dans l'élément `Condition` principal d'une stratégie en utilisant une clé de condition globale `aws:userid`.

```
"Condition": {
    "StringLike": {
      "aws:userId": [
        "AIDACKCEVSQ6C2EXAMPLE",
```

```
        "AROADBQP57FF2AEXAMPLE:role-session-name",
        "AROA1234567890EXAMPLE:*",
        "111122223333"
      ]
    }
  }
```

Un autre exemple dans lequel les ID d'utilisateur puevent être utiles c'est lorsque vous maintenez votre propre base de données (ou autre moyen de stockage) d'un utilisateur IAM ou d'informations sur les rôles. L'ID unique peut fournir un identifiant unique pour chaque utilisateur IAM que vous créez. Cela reste vrai lorsque vous avez des utilisateurs IAM ou des rôles qui réutilisent un nom, comme dans l'exemple précédent.

Présentation des préfixes d'ID uniques

IAM utilise les préfixes suivants pour indiquer le type de ressource auquel chaque ID unique s'applique. Les préfixes peuvent varier en fonction de leur date de création.

Préfixe	Type de ressource
ABIA	AWS STS jeton de service
ACCA	Context-specific justificatif
AGPA	Groupe d'utilisateurs
AIDA	Utilisateur IAM
AIPA	Profil d'instance Amazon EC2
AKIA	Clé d'accès
ANPA	Politique gérée
ANVA	La version d'une politique gérée
APKA	Clé publique
AROA	Role
ASCA	Certificat

Préfixe	Type de ressource
ASIA	Les ID de clé d'accès temporaire (AWS STS) utilisent ce préfixe, mais ne sont uniques qu'en combinaison avec la clé d'accès secrète et le jeton de session.

Obtention de l'identifiant unique

L'ID unique d'une ressource IAM n'est pas disponible dans la console IAM. Pour obtenir l'identifiant unique, vous pouvez utiliser les AWS CLI commandes ou les appels d'API IAM suivants.

AWS CLI:

- get-caller-identity
- get-group
- get-role
- get-user
- get-policy
- get-instance-profile
- get-server-certificate

API IAM

- GetCallerIdentity
- GetGroup
- GetRole
- GetUser
- GetPolicy
- GetInstanceProfile
- GetServerCertificate

IAM et AWS STS quotas

Gestion des identités et des accès AWS (IAM) et AWS Security Token Service (STS) ont des quotas qui limitent la taille des objets. Cela affecte la façon dont vous nommez un objet, le nombre d'objets que vous pouvez créer et le nombre de caractères que vous pouvez utiliser lorsque vous transmettez un objet.

> **Note**
>
> Pour obtenir des informations au niveau du compte sur l'utilisation et les quotas d'IAM, utilisez l'opération GetAccountSummary API ou la commande get-account-summary. https://docs.aws.amazon.com/cli/latest/reference/iam/get-account-summary.html AWS CLI

Exigences relatives aux noms IAM

Les noms IAM ont les exigences et restrictions suivantes :

- Les documents de politique 000A peuvent contenir uniquement les caractères Unicode suivants : tabulation horizontale (U+0009), saut de ligne (U+000A), retour chariot (U+000D) et caractères de la plage U+0020 à U+00FF.
- Les noms d'utilisateurs, de groupes, de rôles, de politiques, de profils d'instance et de certificats de serveur et de chemins doivent être alphanumériques, comprennant les caractères communs suivants : plus (+), égal (=), virgule (,), point (.), arobase (@), trait de soulignement (_) et trait d'union (-). Les noms de chemins doivent commencer et finir par une barre oblique (/).
- Les noms des utilisateurs, des groupes, des rôles et des profils d'instance doivent être uniques au sein du compte. La casse majuscules-minuscules ne compte pas. Par exemple, vous ne pouvez pas créer en même temps des groupes nommés **`ADMINS`** et **`admins`**.
- La valeur d'ID externe qu'un tiers utilise pour endosser un rôle doit avoir au minimum 2 caractères et au maximum 1 224 caractères. La valeur doit être alphanumérique sans espaces. Elle peut également inclure les symboles suivants : signe plus (+), signe égal (=), virgule (,), point (.), arobase (@), deux points (:), barre oblique (/) et tiret (-). Pour plus d'informations sur l'ID externe, consultez Accès à Comptes AWS appartenant à des tiers.
- Les noms de politique pour les politiques en ligne doivent être uniques pour l'utilisateur, le groupe ou le rôle auquel ils sont intégrés. Les noms peuvent contenir tous les caractères latins de base (ASCII), à l'exception des caractères réservés suivants : barre oblique inversée (\), barre oblique

(/), astérisque (*), point d'interrogation (?) et espace. Ces caractères sont réservés conformément à la norme RFC 3986, section 2.2.

- Les mots de passe des utilisateurs (profils de connexion) peuvent contenir tous les caractères latins (ASCII) de base.
- Compte AWS Les alias d'identification doivent être uniques pour tous les AWS produits et doivent être alphanumériques conformément aux conventions de dénomination DNS. Un alias doit être en minuscules, ne doit pas commencer ou se terminer par un trait d'union, ne peut pas contenir deux traits d'union consécutifs et ne peut pas être un numéro à 12 chiffres.

Pour une liste des caractères latins de base (ASCII), consultez la table de code ASCII (Library of Congress Basic Latin).

Quotas d'objet IAM

Les quotas, également appelés limites dans AWS, sont les valeurs maximales pour les ressources, les actions et les éléments de votre Compte AWS. Utilisez Service Quotas pour gérer vos quotas IAM.

Pour obtenir la liste des points de terminaison et des quotas de service IAM, consultez Points de terminaison et quotas de service Gestion des identités et des accès AWS (français non garanti) dans Références générales AWS.

Pour demander une augmentation de quota

1. Suivez la procédure de connexion adaptée à votre type d'utilisateur, comme décrit dans la rubrique Comment se connecter AWS à du Guide de l'AWS Sign-In utilisateur pour vous connecter au Console de gestion AWS.
2. Ouvrez la console Service Quotas.
3. Dans le panneau de navigation, choisissez Services AWS .
4. Dans la barre de navigation, sélectionnez la région US East (N. Virginia). Ensuite, recherchez **`IAM`**.
5. Sélectionnez Gestion des identités et des accès AWS (IAM), choisissez un quota et suivez les instructions pour demander une augmentation de quota.

Pour de plus amples informations, veuillez consulter Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

Pour voir un exemple de demande d'augmentation de quota IAM à l'aide de la console Service Quotas, regardez la vidéo suivante.

Demandez une augmentation de quota IAM à l'aide de la console Service Quotas.

Vous pouvez demander une augmentation des quotas par défaut pour les quotas ajustables IAM. Les demandes jusqu'à maximum quota sont automatiquement approuvées et terminées en quelques minutes.

Le tableau suivant répertorie les ressources pour lesquelles les augmentations de quotas peuvent être automatiquement approuvées.

Ressource	Quota par défaut	Quota maximal
Politiques gérées par le client par compte	1 500	10 000
Groupes par compte	300	500
Profils d'instances par compte	1 000	10 000
Politiques gérées par rôle	20	25
Politiques gérées par utilisateur	10	20
Politiques gérées par groupe	10	10
Longueur de la politique d'approbation du rôle	2048 caractères	8192 caractères
Rôles par compte	1 000	10 000
Certificats de serveur par compte	20	20
OpenId connecter les fournisseurs par compte	100	700

Quotas de l'IAM Access Analyzer

Pour obtenir la liste des points de terminaison et des quotas de service de l'analyseur d'accès IAM, consultez Points de terminaison et quotas de l'analyseur d'accès IAM (français non garanti) dans Références générales AWS.

Quotas des rôles Anywhere IAM

Pour obtenir la liste des points de terminaison et des quotas de service des rôles Anywhere IAM, consultez Points de terminaison et quotas de service des rôles Anywhere Gestion des identités et des accès AWS (français non garanti) dans Références générales AWS.

Quotas de requête STS

Le AWS Security Token Service (AWS STS) applique les quotas de demandes suivants.

Pour les AWS STS demandes effectuées à l'aide AWS d'informations d'identification, le quota de demandes par défaut est de 600 demandes par seconde, par compte et par région. Les AWS STS opérations suivantes se partagent ce quota :

- AssumeRole
- DecodeAuthorizationMessage
- GetAccessKeyInfo
- GetCallerIdentity
- GetFederationToken
- GetSessionToken

> **Note**
>
> Les demandes émanant des responsables de AWS service, telles que celles utilisées pour assumer des rôles à utiliser dans un AWS service, ne consomment pas le quota de requêtes STS par seconde dans vos comptes. AWS STS

Par exemple, si un Compte AWS utilisateur effectue 100 GetCallerIdentity demandes par seconde et 100 AssumeRole appels par seconde dans la même région, ce compte consomme 200 des 600 requêtes STS disponibles par seconde pour cette région.

Pour les AssumeRole demandes intercomptes, seul le compte à l'origine de la AssumeRole demande a un impact sur le quota STS. Le compte cible n'a pas de quota consommé.

Pour demander une augmentation des quotas de requêtes STS, veuillez ouvrir un ticket auprès de l'assistance AWS .

> **Note**
>
> Avec les prochaines modifications apportées au point de terminaison AWS STS global (`https://sts.amazonaws.com`), les demandes adressées au point de terminaison global ne partageront pas de quota de demandes par seconde (RPS) avec les points de terminaison AWS STS régionaux dans les régions qui sont activées par défaut. Lorsqu'une demande adressée au terminal AWS STS mondial provient d'une seule région, elle est prise en compte dans le quota RPS du point de terminaison mondial. Toutefois, lorsque les demandes proviennent de plusieurs régions, chaque région supplémentaire recevra son propre quota RPS indépendant. Pour plus d'informations sur les modifications AWS STS globales des points de terminaison, consultezAWS STS modifications du point de terminaison global.

Limites des caractères d'IAM et de STS

Le nombre maximal de caractères et les limites de taille pour IAM et AWS STS sont les suivants. Vous ne pouvez pas demander une augmentation pour les limites suivantes.

Description	Limite
Alias pour un Compte AWS identifiant	3–63 caractères
Pour les politiques en ligne	Vous pouvez ajouter autant de politiques en ligne que désiré à un utilisateur, un rôle ou un groupe IAM. Toutefois, la taille totale de la politique agrégée (la somme de toutes les politiques en ligne) par entité ne peut pas dépasser les limites suivantes : • La taille de la politique d'utilisateur ne peut pas dépasser 2 048 caractères.

Description	Limite
	• La taille de la politique de rôle ne peut pas dépasser 10 240 caractères. • La taille de la politique de groupe ne peut pas dépasser 5 120 caractères. **Note** IAM ne compte pas les espaces lors du calcul de la taille d'une politique au regard de ces limites.
Pour les politiques gérées par le client	• La taille de chaque politique gérée ne peut pas dépasser 6 144 caractères. **Note** IAM ne compte pas les espaces lors du calcul de la taille d'une politique au regard de cette limite.
Nom du groupe	128 caractères
Nom du profil d'instance	128 caractères
Mot de passe pour un profil de connexion	1–128 caractères
Chemin	512 caractères
Nom de la politique	128 caractères

Description	Limite
Nom de rôle	64 caractères **Important** Si vous avez l'intention d'utiliser un rôle avec la fonctionnalité Changer de rôle dans le Console de gestion AWS, alors la combinaison `Path` ne `RoleName` peut pas dépasser 64 caractères.
Durée de la session de rôle	12 heures Lorsque vous assumez un rôle depuis l'API AWS CLI or, vous pouvez utiliser le paramètre `duration-seconds` CLI ou le paramètre `DurationSeconds` API pour demander une session de rôle plus longue. Vous pouvez spécifier une valeur comprise entre 900 secondes (15 minutes) et la durée de session maximale pour le rôle, qui peut varier de 1 à 12 heures. Si vous ne spécifiez pas de valeur pour le paramètre `DurationSeconds` , vos informations d'identification de sécurité sont valides pendant une heure. Les utilisateurs IAM qui changent de rôle dans la console se voient accorder la durée de session maximale ou le temps restant dans la session de l'utilisa teur IAM, selon la durée la plus courte. Le paramètre de durée maximale de session ne limite pas les sessions endossées par les services AWS . Pour savoir comment afficher la valeur maximale pour votre rôle, veuillez consulter Mettre à jour la durée de session maximale pour un rôle.

Description	Limite
Nom de la session de rôle	64 caractères
Politiques de session du rôle	• La taille du document de politique JSON transmis et tous les caractères ARN de politique gérée transmis combinés ne peut pas dépasser 2 048 caractères. • Vous pouvez transmettre un maximum de 10 ARN de politique gérée lorsque vous créez une session. • Vous ne pouvez transmettre qu'un seul document de politique JSON lorsque vous créez par programmation une session temporaire pour un rôle ou un utilisateur principal AWS STS fédéré. • En outre, les politiques de session et les balises de session augmentent la taille du jeton de session, qui est soumise à une limite distincte. L'élément de `SessionTokenUtilization` réponse indique le pourcentage de la limite de taille utilisé par le jeton. L'élément de `PackedPolicySize` réponse est obsolète. Utilisez `SessionTokenUtilization` à la place. • Nous vous recommandons de transmettre les politiques de session à l'aide de l' AWS API AWS CLI or. Console de gestion AWS Cela peut augmenter la taille du jeton de session en ajoutant des informations de session.

Description	Limite
Balises de session du rôle	• Les balises de session doivent respecter la limite de la clé de balise de 128 caractère s et la limite de la valeur de balise de 256 caractères. • Vous pouvez transmettre jusqu'à 50 balises de session. • Les politiques de session et les balises de session s'ajoutent à la taille du jeton de session, qui est soumise à une limite distincte. Vous pouvez transmettre des balises de session à l'aide de l' AWS API AWS CLI or. L'élément de `SessionTo kenUtilization` réponse indique le pourcentage de la limite de taille utilisé par le jeton. L'élément de `PackedPolicySize` réponse est obsolète. Utilisez `SessionTo kenUtilization` à la place.
Réponse d'authentification SAML codée en base64	100 000 caractères Cette limite de caractères s'applique à l'opérati on CLI `assume-role-with-saml` ou d'API `AssumeRoleWithSAML`.
Clé de balise	128 caractères Cette limite de caractères s'applique aux balises des ressources IAM et des balises de session.

Description	Limite
Valeur de balise	256 caractères Cette limite de caractères s'applique aux balises des ressources IAM et des balises de session. Les valeurs de balise peuvent être vides, ce qui signifie que les valeurs de balise peuvent ne comporter aucun caractère.
ID uniques créés par IAM	128 caractères Par exemple : • ID d'utilisateur commençant par `AIDA` • ID de groupe commençant par `AGPA` • ID de rôle commençant par `AROA` • ID de politique gérée commençant par `ANPA` • ID de stratégie gérée commençant par `ASCA` **Note** Cette liste n'est pas exhaustive. Elle ne garantit pas non plus que les ID d'un certain type commencent uniquement par la combinaison de lettres indiquée.
Nom utilisateur	64 caractères

Prise en charge des points de terminaison double pile

Les points de terminaison à double pile permettent aux clients de communiquer avec les AWS services en utilisant l'une IPv4 ou IPv6 l'autre des adresses. IAM et AWS STS fournissent tous deux un support de point de terminaison à double pile. Pour plus d'informations sur la façon de configurer

votre VPC pour le mode double pile, consultez le guide de l'utilisateur Amazon VPC sur la prise en IPv6 charge de votre VPC.

Si vous utilisez des politiques IAM qui incluent les clés de `aws:vpcSourceIp` condition `aws:sourceIp` or, vous devez mettre à jour ces politiques pour prendre en charge les IPv6 clients. Pour plus d'informations sur l' IPv6 adressage de votre compte VPCs, consultez la section Adressage IP de votre réseau VPCs et de vos sous-réseaux dans le guide de l'utilisateur Amazon VPC.

Prise en charge des points de terminaison double pile IAM

IAM fournit un point de terminaison public à double pile qui prend en charge à la fois les clients IPv4 et les clients IPv6 . Le point de terminaison public à double pile IAM est également accessible en privé depuis votre cloud privé virtuel (VPC) à l'aide de. AWS PrivateLink Pour plus d'informations sur la création de points de terminaison d'un VPC d'interface privée pour IAM, consultez Création d'un point de terminaison de VPC pour IAM.

Le point de terminaison public à double pile IAM est. `https://iam.global.api.aws`

Le point de terminaison public IAM situé à l'adresse `https://iam.amazonaws.com`, contrairement au point de terminaison public à double pile, ne prend en charge que IPv4 les clients. Lorsqu'il est accessible de manière privée depuis votre VPC à l'aide de AWS PrivateLink, le point de terminaison public IAM peut prendre en charge à la fois les IPv4 clients et les clients. IPv6

AWS STS prise en charge des terminaux à double pile

AWS STS fournit des points de terminaison régionaux à double pile qui prennent en charge à la fois IPv4 les IPv6 clients. Les points de terminaison régionaux AWS STS à double pile sont également accessibles en privé depuis votre cloud privé virtuel (VPC) à l'aide de. AWS PrivateLink Pour plus d'informations sur la création de points de terminaison VPC d'interface privée pour AWS STS, consultez. Créez un point de terminaison VPC pour AWS STS

Le tableau suivant indique les points de terminaison régionaux AWS STS à double pile par partition :

Partition	URL du point de terminaison à double pile
AWS Régions commerciales	Points de terminaison habituels : `https://sts. <region>.api.aws` Points de terminaison FIPS : `https://sts-fips.<region>.api.aws`

Partition	URL du point de terminaison à double pile
AWS GovCloud (US) Régions	Points de terminaison habituels : `https://sts.<region>.api.aws`
Régions Chine	Points de terminaison habituels : `https://sts.<region>.api.amazonwebservices.com.cn`

> **Note**
>
> Les points de terminaison FIPS ne s'appliquent qu'aux régions des États-Unis et du Canada.

Limitations

Les limites suivantes s'appliquent à la prise en charge des terminaux AWS STS à double pile :

- IPv6 les clients ne sont pas pris en charge sur le point de terminaison globalhttps://sts.amazonaws.com. Vous devez utiliser des points de terminaison régionaux à double pile pour le support IPv6 client.
- IPv6-seuls les points de terminaison AWS STS VPC ne sont pas pris en charge. Les points de terminaison VPC peuvent être configurés pour IPv4 une connectivité à double pile.

Points de terminaison de VPC d'Interface

Si vous utilisez Amazon Virtual Private Cloud (Amazon VPC) pour héberger vos AWS ressources, vous pouvez établir une connexion privée entre votre VPC et Gestion des identités et des accès AWS (IAM) ou AWS Security Token Service ().AWS STS Vous pouvez utiliser cette connexion pour activer IAM ou AWS STS pour communiquer avec vos ressources dans votre VPC sans passer par l'Internet public.

Amazon VPC est un AWS service que vous pouvez utiliser pour lancer AWS des ressources dans un réseau virtuel que vous définissez. Avec un VPC, vous contrôlez des paramètres réseau, tels que la plage d'adresses IP, les sous-réseaux, les tables de routage et les passerelles réseau. Pour connecter votre VPC à IAM ou AWS STS, vous définissez un point de terminaison VPC d'interface pour chaque service. Le terminal fournit une connectivité fiable et évolutive à IAM ou AWS STS sans nécessiter de passerelle Internet, d'instance de traduction d'adresses réseau (NAT) ou de connexion

VPN. Pour de plus amples informations, veuillez consulter Qu'est-ce qu'Amazon VPC ? dans le Guide de l'utilisateur Amazon VPC.

Interface Les points de terminaison VPC sont alimentés par AWS PrivateLink une AWS technologie qui permet une communication privée entre les AWS services à l'aide d'une interface réseau élastique avec des adresses IP privées. Pour plus d'informations, consultez AWS PrivateLink la section AWS Services.

Les informations suivantes sont destinées aux utilisateurs d'Amazon VPC. Pour de plus amples informations, veuillez consulter Mise en route avec Amazon VPC dans le Guide de l'utilisateur Amazon VPC.

Rubriques

- Disponibilité des points de terminaison d'un VPC
- Création d'un point de terminaison de VPC pour IAM
- Créez un point de terminaison VPC pour AWS STS
- Créez un point de terminaison VPC pour AWS STS Découverte de l'OIDC
- Contrôlez l'accès à AWS STS avec des politiques relatives aux terminaux VPC
- Utiliser les clés de condition VPC pour contrôler l'accès fédéré

Disponibilité des points de terminaison d'un VPC

> **Important**
>
> Les points de terminaison d'un VPC d'interface pour IAM ne peuvent être créés que dans la région où se trouve le plan de contrôle IAM. Si votre VPC se trouve dans une région différente de celle du plan de contrôle IAM, vous devez l'utiliser AWS Transit Gateway pour autoriser l'accès au point de terminaison VPC de l'interface IAM depuis une autre région. Pour plus d'informations, consultez Création d'un point de terminaison de VPC pour IAM.

IAM prend actuellement en charge les points de terminaison d'un VPC dans les régions suivantes :

- USA Est (Virginie du Nord)
- Chine (Pékin)
- AWS GovCloud (US-West)

AWS STS prend actuellement en charge les points de terminaison VPC dans les régions suivantes :

- USA Est (Virginie du Nord)
- USA Est (Ohio)
- USA Ouest (Californie du Nord)
- USA Ouest (Oregon)
- Afrique (Le Cap)
- Asie-Pacifique (Hong Kong)
- Asie-Pacifique (Hyderabad)
- Asie-Pacifique (Jakarta)
- Asie-Pacifique (Melbourne)
- Asie-Pacifique (Mumbai)
- Asie-Pacifique (Osaka)
- Asia Pacific (Seoul)
- Asie-Pacifique (Singapour)
- Asie-Pacifique (Sydney)
- Asie-Pacifique (Tokyo)
- Canada (Centre)
- Canada Ouest (Calgary)
- Chine (Pékin)
- China (Ningxia)
- Europe (Francfort)
- Europe (Irlande)
- Europe (Londres)
- Europe (Milan)
- Europe (Paris)
- Europe (Espagne)
- Europe (Stockholm)
- Europe (Zurich)
- Israël (Tel Aviv)

- Middle East (Bahrain)
- Moyen-Orient (EAU)
- Amérique du Sud (São Paulo)
- AWS GovCloud (US-East)
- AWS GovCloud (US-West)

Création d'un point de terminaison de VPC pour IAM

Pour commencer à utiliser IAM avec votre VPC, créez un point de terminaison de VPC d'interface pour IAM. Pour plus d'informations, consultez la section Accès à un AWS service à l'aide d'un point de terminaison VPC d'interface dans le guide de l'utilisateur Amazon VPC.

Les points de terminaison d'un VPC d'interface pour IAM ne peuvent être créés que dans la région où se trouve le plan de contrôle IAM. Sur le plan commercial Régions AWS, le plan de contrôle IAM est situé dans la région de l'est des États-Unis (Virginie du Nord) (us-east-1). Le nom du service de point de terminaison VPC de l' AWS PrivateLink interface pour IAM est. `com.amazonaws.iam` Pour obtenir la liste des points Régions AWS de terminaison VPC compatibles avec IAM, consultez. Disponibilité des points de terminaison d'un VPC

Si votre VPC se trouve dans une région différente de celle du plan de contrôle IAM, vous devez utiliser AWS Transit Gateway pour autoriser l'accès au point de terminaison de VPC d'interface IAM depuis une autre région.

Pour accéder à un point de terminaison VPC d'interface IAM à partir d'un VPC d'une autre région en utilisant AWS Transit Gateway

1. Créez une passerelle de transit ou utilisez une passerelle de transit existante pour interconnecter vos clouds privés virtuels (VPCs). Une passerelle de transit est requise pour chaque région. Pour plus d'informations, consultez la rubrique Création d'une passerelle de transit du Guide AWS Transit Gateway .
2. Créez des attachements VPC à la passerelle de transit pour connecter chaque VPC à la passerelle de transit. Pour plus d'informations, consultez la rubrique Création d'un attachement de la passerelle de transit vers un VPC du Guide AWS Transit Gateway .
3. Créez une pièce jointe d'appairage VPC pour une passerelle de transit pour acheminer le trafic entre pairs. VPCs Pour plus d'informations, consultez Création d'un attachement d'apparaige dans le Guide AWS Transit Gateway .

> **Note**
>
> Les connexions d'appairage VPC peuvent également acheminer le trafic entre des connexions d'appairage VPCs, mais cette méthode n'est pas adaptée à un grand nombre de. VPCs Au lieu de l'apparaige de VPC, nous recommandons les attachements d'apparaige AWS Transit Gateway qui améliorent la gestion des VPC et des réseaux sur site par le biais d'un hub central évolutif. Pour de plus amples informations sur les connexions d'appairage de VPC, consultez Travailler avec des connexions d'apparaige de VPC dans le Guide d'apparaige de VPC Amazon.

Créez un point de terminaison VPC pour AWS STS

Pour commencer à l'utiliser AWS STS avec votre VPC, créez une interface pour laquelle le point de terminaison du VPC. AWS STS Pour plus d'informations, consultez Accéder à un AWS service à l'aide d'un point de terminaison VPC d'interface dans le guide de l'utilisateur Amazon VPC.

Après avoir créé le point de terminaison VPC, vous devez utiliser le point de terminaison régional correspondant pour envoyer vos AWS STS demandes. AWS STS vous recommande d'utiliser à la fois les `setEndpoint` méthodes `setRegion` et pour passer des appels vers un point de terminaison régional. Vous pouvez utiliser la méthode `setRegion` de manière autonome pour répondre à des régions activées manuellement, par exemple Asie-Pacifique (Hong Kong). Dans ce cas, les appels sont dirigés vers le point de terminaison régional STS. Pour savoir comment activer manuellement une région, veuillez consulter Gestion des régions AWS dans le Références générales AWS. Si vous utilisez la méthode `setRegion` uniquement pour les régions activées par défaut, les appels sont dirigés vers le point de terminaison global `https://sts.amazonaws.com` .

Lorsque vous utilisez des points de terminaison régionaux, AWS STS appelez d'autres AWS services à l'aide de points de terminaison publics ou de points de terminaison VPC à interface privée, selon le type utilisé. Supposons, par exemple, que vous ayez créé une interface pour laquelle vous avez créé un point de terminaison VPC AWS STS et que vous ayez déjà demandé des informations d'identification temporaires AWS STS auprès de ressources situées dans votre VPC. Dans ce cas, ces informations d'identification commencent à circuler via le point de terminaison VPC d'interface par défaut. Pour plus d'informations sur l'envoi de demandes régionales à l'aide de AWS STS, consultezGérer AWS STS dans un Région AWS.

Créez un point de terminaison VPC pour AWS STS Découverte de l'OIDC

AWS Security Token Service (AWS STS) La découverte OIDC est une paire de points de terminaison publics qui fournissent les métadonnées et les clés de vérification dont les parties ont besoin pour valider les jetons Web JSON (JWT) émis par la fédération d'identité sortante.

Lorsque vous activez la fédération des identités sortantes, AWS STS publie une URL d'émetteur spécifique au compte. Cette URL héberge deux points de terminaison de découverte OpenID Connect (OIDC). Les parties utilisatrices récupèrent ces points de terminaison pour obtenir les métadonnées et les clés publiques dont elles ont besoin pour vérifier les JWT qui émettent : GetWebIdentityToken

- `{issuer_url}/.well-known/openid-configuration`— Le document de découverte OIDC
- `{issuer_url}/.well-known/jwks.json`— Le jeu de clés Web JSON (JWKS) qui contient les clés de vérification de la signature du jeton

Par défaut, l'accès à ces points de terminaison depuis votre VPC nécessite une passerelle Internet, un périphérique NAT, une connexion VPN ou une connexion. Pour conserver ce trafic sur le AWS réseau, créez plutôt un point de terminaison VPC d'interface pour le point de terminaison de découverte OIDC STS.

> **Note**
>
> Le point de terminaison de découverte OIDC du STS dessert uniquement les deux points de terminaison répertoriés ci-dessus. Il ne sert à aucune opération AWS STS d'API. Pour appeler `GetWebIdentityToken` ou `AssumeRole` effectuer toute autre AWS STS opération en privé, utilisez le service `com.amazonaws.region.sts` Endpoint. Consultez Créez un point de terminaison VPC pour AWS STS.

Choisir le bon service de point de terminaison

La fédération d'identité sortante utilise deux chemins réseau distincts. La plupart des charges de travail qui génèrent et vérifient des jetons nécessitent les deux services de point de terminaison.

Sous-tâche	Service de terminaux	Authentification
Créer un JWT () `sts:GetWebIdentityToken`	`com.amazonaws.`*`region`*`.sts`	AWS SIGv4
Lisez `/.well-known/openid-configuration`	`com.amazonaws.`*`region`*`.sts-oidc`	Aucun (non authentifié)
Lisez `/.well-known/jwks.json`	`com.amazonaws.`*`region`*`.sts-oidc`	Aucun (non authentifié)

Vous n'avez pas besoin du point de terminaison de découverte STS OIDC si la seule partie qui vérifie vos jetons est un service externe externe. AWS Ce service récupère l'URL de votre émetteur sur l'Internet public et son trafic n'entre pas dans votre VPC.

Considérations

Tenez compte des points suivants avant de créer un point de terminaison VPC d'interface pour la découverte STS OIDC.

- Les politiques de point de terminaison d'un VPC ne sont pas prises en charge. Les deux API de découverte fournissent des données accessibles au public, et le point de terminaison de découverte OIDC STS n'applique pas les politiques de point de terminaison VPC. Utilisez les groupes de sécurité et le placement des sous-réseaux pour contrôler quelles ressources peuvent atteindre le terminal.
- Seuls deux chemins sont desservis. Le point de terminaison accepte `GET` les requêtes HTTPS destinées à `/.well-known/openid-configuration` et `/.well-known/jwks.json` uniquement. Tous les autres chemins et méthodes renvoient une erreur.

Création d'un point de terminaison VPC d'interface pour la découverte STS OIDC

Vous pouvez créer un point de terminaison VPC pour le point de terminaison STS OIDC Discovery à l'aide de la console Amazon Virtual Private Cloud ou du. AWS Command Line Interface Pour plus d'informations, consultez Accéder à un AWS service à l'aide d'un point de terminaison VPC d'interface dans le guide de l'utilisateur Amazon VPC.

Créez un point de terminaison VPC pour la découverte STS OIDC en utilisant le nom de service suivant :

- `com.amazonaws.region.sts-oidc`

Si vous activez le DNS privé pour le terminal, vous pouvez envoyer des requêtes aux `sts-oidc.region.amazonaws.com` interfaces réseau du terminal au sein de votre VPC, qui seront résolues vers ces interfaces.

Envoi de demandes via le terminal

Lorsqu'une partie utilisatrice de votre VPC doit vérifier un JWT, elle récupère le document de découverte OIDC et le JWKS à partir de l'URL de l'émetteur. Le nom d'hôte de l'émetteur (par exemple`uuid.tokens.sts.global.api.aws`) utilise un domaine différent de celui du point de terminaison (`sts-oidc.region.amazonaws.com`), il n'est donc pas résolu par défaut vers le point de terminaison VPC.

Pour acheminer les demandes concernant le nom d'hôte de l'émetteur via le point de terminaison, créez une zone hébergée privée Amazon Route 53 pour le nom d'hôte de votre émetteur et ajoutez un enregistrement d'alias qui pointe vers le point de terminaison du VPC. Une fois que vous avez créé la zone hébergée, les ressources du VPC réfèrent le nom d'hôte de l'émetteur aux interfaces réseau du terminal. Les bibliothèques OIDC standard peuvent ensuite récupérer les documents de découverte sans modifier le code. Pour de plus amples informations sur la création d'une zone hébergée privée, veuillez consulter Utilisation de zones hébergées privées dans le Guide du développeur Amazon Route 53.

> **Note**
>
> Le service de point de terminaison est régional, mais un seul point de terminaison répond aux demandes de découverte pour tout UUID émetteur de la partition. Créez des points de terminaison dans les régions où fonctionnent vos vérificateurs ; vous n'en avez pas besoin d'un par compte émetteur.

Contrôlez l'accès à AWS STS avec des politiques relatives aux terminaux VPC

Lorsque vous créez un point de terminaison VPC d'interface pour AWS Security Token Service (AWS STS), vous pouvez y associer une politique de point de terminaison. La politique contrôle quels responsables peuvent utiliser le point de terminaison et quelles AWS STS actions ils peuvent effectuer. Si vous ne joignez aucune politique, le point de terminaison utilise la politique par défaut qui autorise un accès illimité à toutes les AWS STS actions pour tous les principaux acteurs.

Les politiques relatives aux terminaux VPC n'accordent pas d'autorisations à elles seules. Elles constituent une limite supplémentaire qui fonctionne parallèlement à d'autres politiques. La politique des terminaux et les politiques applicables à l'appelant doivent toutes deux autoriser une demande pour qu'elle aboutisse.

Pour plus d'informations sur les politiques relatives aux terminaux VPC, consultez la section Contrôler l'accès aux points de terminaison VPC à l'aide des politiques relatives aux terminaux dans le guide de l'utilisateur Amazon VPC.

Rubriques

- Politique de point de terminaison d'un VPC par défaut
- Considérations importantes pour AWS STS Politiques de point de terminaison d'un VPC
- Clés de condition disponibles pour AWS STS Politiques de point de terminaison d'un VPC
- Exemple : Tout autoriser AWS STS actions pour votre organisation
- Exemple : Tout autoriser AWS STS actions pour des comptes spécifiques
- Exemple : Restreindre à des données spécifiques AWS STS actions
- Exemple : Refuser l'accès non organisationnel tout en autorisant la fédération

Politique de point de terminaison d'un VPC par défaut

Si vous ne joignez pas de stratégie personnalisée lorsque vous créez le point de terminaison, AWS associez la stratégie par défaut suivante. Cette politique permet un accès illimité au terminal.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": "*",
            "Resource": "*"
        }
    ]
}
```

Pour limiter l'accès au terminal, joignez une politique de point de terminaison personnalisée.

Considérations importantes pour AWS STS Politiques de point de terminaison d'un VPC

AWS STS gère les demandes de deux types d'appelants fondamentalement différents. Votre politique de point de terminaison VPC doit prendre en compte les deux types pour éviter de bloquer involontairement les demandes légitimes.

Mandants authentifiés AWS

Utilisateurs IAM et rôles IAM qui signent des demandes avec AWS Signature Version 4 (Sigv4). Ces appelants disposent de clés de condition standard telles que`aws:PrincipalOrgID`,`aws:PrincipalAccount`, et `aws:PrincipalArn` sont disponibles dans le contexte de la demande.

Appels fédérés

Principaux SAML 2.0 et OpenID Connect (OIDC) qui appellent ou. `AssumeRoleWithSAML` `AssumeRoleWithWebIdentity` Ces appelants s'authentifient à l'aide d'assertions SAML ou de jetons Web JSON (JWT), et non de signatures Sigv4. Comme ils n'ont aucune AWS identité au moment de la demande, le contexte de la demande n'inclut pas de clés de condition basées sur le principe`aws:PrincipalOrgID`, telles que`aws:PrincipalAccount`, et. `aws:PrincipalArn`

> **Important**
>
> Si votre politique de point de terminaison VPC repose uniquement sur l'`aws:PrincipalOrgID`autorisation d'accès, les `AssumeRoleWithWebIdentity` appels `AssumeRoleWithSAML` fédérés sont implicitement refusés car la clé de condition est absente pour les utilisateurs non principaux.AWS

Comment ? AWS STS évalue les politiques relatives aux terminaux VPC pour les appelants fédérés

Lorsqu'un appelant fédéré appelle `AssumeRoleWithSAML` ou `AssumeRoleWithWebIdentity` via un point de terminaison VPC, les règles suivantes s'appliquent :

- L'appelant n'est pas un AWS mandant. Les clés de condition telles que `aws:PrincipalOrgIDaws:PrincipalAccount`, et ne `aws:PrincipalArn` sont pas disponibles dans le contexte de la demande.
- Les appelants fédérés ne disposent pas de la clé de condition `aws:PrincipalIsAWSService` dans le contexte de la demande.
- Le rôle assumé est celui d'une AWS ressource. Resource-based les clés de condition telles que `aws:ResourceOrgID` et `aws:ResourceAccount` sont disponibles et font référence au rôle cible.
- La politique de confiance du rôle reste la principale porte d'autorisation pour l'accès fédéré. La politique de point de terminaison VPC fournit une limite supplémentaire au niveau du réseau.

Nous vous recommandons d'utiliser `aws:ResourceOrgID` ou `aws:ResourceAccount` de rédiger des déclarations de politique de point de terminaison VPC qui doivent s'appliquer aux appelants fédérés, car ces appelants ne disposent pas de clés de condition basées sur le principal.

Clés de condition disponibles pour AWS STS Politiques de point de terminaison d'un VPC

Le tableau suivant présente les clés de condition couramment utilisées et leur disponibilité dans le contexte de la demande pour chaque type d'appelant lors de l'envoi de demandes via un point de terminaison AWS STS VPC. Des clés de condition supplémentaires sont disponibles en plus de celles répertoriées ici.

Disponibilité des clés de condition par type d'appelant

Clé de condition	Mandants authentifiés AWS	Appels fédérés	Description
`aws:PrincipalOrgID`	Oui	Non	L'identifiant de l'organisation du principal appelant
`aws:PrincipalAccount`	Oui	Non	L'identifiant de compte du principal appelant

Clé de condition	Mandants authentifiés AWS	Appels fédérés	Description
`aws:PrincipalArn`	Oui	Non	L'ARN du principal appelant
`aws:PrincipalIsAWSService`	Oui (la valeur est évaluée à faux)	Non (la clé est absente)	Si l'appelant est un directeur AWS de service
`aws:ResourceOrgID`	Oui	Oui	L'identifiant de l'organisation du compte propriétaire de la ressource demandée
`aws:ResourceAccount`	Oui	Oui	L'ID de compte propriétaire de la ressource demandée

> **Note**
>
> Pour les AWS principaux authentifiés (utilisateurs et rôles IAM), `aws:PrincipalIsAWSService` est présent dans le contexte de la demande et est évalué à faux. Pour les appelants fédérés, cette clé est totalement absente du contexte de la demande. Une condition qui vérifie qu'`"Bool": {"aws:PrincipalIsAWSService": "false"}`elle ne correspond pas aux appelants fédérés car la clé n'est pas présente.

Exemple : Tout autoriser AWS STS actions pour votre organisation

La politique de point de terminaison suivante limite votre point de terminaison AWS STS VPC à votre organisation tout en prenant en charge l'accès fédéré. Il autorise toutes les AWS STS actions pour les principaux authentifiés de votre organisation et permet séparément aux appelants fédérés d'assumer des rôles au sein de votre organisation. Les appelants fédérés (`AssumeRoleWithSAML`et`AssumeRoleWithWebIdentity`) ont besoin d'une déclaration distincte car ils n'en ont pas `aws:PrincipalOrgID` dans le contexte de la demande.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "AllowOrganizationPrincipals",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "sts:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": "o-exampleorgid"
                }
            }
        },
        {
            "Sid": "AllowFederatedAssumeRole",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "sts:AssumeRoleWithSAML",
                "sts:AssumeRoleWithWebIdentity"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceOrgID": "o-exampleorgid"
                }
            }
        }
    ]
}
```

La première instruction utilise `"Principal": {"AWS": "*"}` with `aws:PrincipalOrgID` pour autoriser les AWS principaux authentifiés de votre organisation. La deuxième instruction permet `"Principal": "*"` de faire correspondre les appelants fédérés et limite le rôle cible à l'utilisation de votre organisation. `aws:ResourceOrgID` La politique de confiance du rôle reste le principal contrôle pour lequel les identités fédérées peuvent assumer quels rôles.

Pour plus d'informations sur la mise en œuvre des contrôles du périmètre du réseau, consultez la section Création d'un périmètre de données AWS et les exemples de politiques de périmètre de données sur le GitHub site Web.

Exemple : Tout autoriser AWS STS actions pour des comptes spécifiques

La politique de point de terminaison suivante autorise toutes les AWS STS actions pour les responsables de comptes spécifiques. Utilisez cette politique lorsque vos comptes ne font pas partie d'une organisation. Les appelants fédérés sont autorisés dans une déclaration distincte car ils n'en ont pas `aws:PrincipalAccount` dans le contexte de la demande.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSpecificAccountPrincipals",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "sts:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalAccount": [
                        "111122223333",
                        "444455556666"
                    ]
                }
            }
        },
        {
            "Sid": "AllowFederatedAssumeRole",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "sts:AssumeRoleWithSAML",
                "sts:AssumeRoleWithWebIdentity"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": [
                        "111122223333",
                        "444455556666"
                    ]
                }
```

```
            }
        }
    ]
}
```

La première instruction utilise `"Principal": {"AWS": "*"}` with `aws:PrincipalAccount` pour autoriser les AWS principaux authentifiés à partir des comptes spécifiés. La deuxième instruction permet `"Principal": "*"` de faire correspondre les appelants fédérés et limite le rôle cible aux mêmes comptes que ceux qui utilisent. `aws:ResourceAccount` La politique de confiance du rôle reste le principal contrôle pour lequel les identités fédérées peuvent assumer quels rôles.

Exemple : Restreindre à des données spécifiques AWS STS actions

La politique de point de terminaison suivante n'autorise que `AssumeRole` les principaux authentifiés et `AssumeRoleWithSAML` les appelants fédérés, tous deux limités aux rôles au sein de votre organisation.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAssumeRoleByOrgPrincipals",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "sts:AssumeRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": "o-exampleorgid"
                }
            }
        },
        {
            "Sid": "AllowSAMLFederation",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "sts:AssumeRoleWithSAML",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
```

```
                    "aws:ResourceOrgID": "o-exampleorgid"
                }
            }
        }
    ]
}
```

Cette politique bloque d'autres AWS STS actions telles que `GetCallerIdentityGetSessionToken`, et `AssumeRoleWithWebIdentity` via ce point de terminaison. Ajustez les `Action` éléments en fonction de vos besoins.

Exemple : Refuser l'accès non organisationnel tout en autorisant la fédération

La politique de point de terminaison suivante utilise un refus explicite pour bloquer les principaux authentifiés extérieurs à votre organisation, tout en préservant l'accès pour les appelants fédérés. Cette approche commence par une autorisation générale et ajoute une déclaration de refus ciblée.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAll",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "*",
            "Resource": "*"
        },
        {
            "Sid": "DenyNonOrgPrincipals",
            "Effect": "Deny",
            "Principal": {
                "AWS": "*"
            },
            "Action": "sts:*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalOrgID": "o-exampleorgid"
                }
            }
        }
    ]
```

```
}
```

L'instruction de refus utilise`"Principal": {"AWS": "*"}`, ce qui l'étend uniquement aux AWS principaux authentifiés. Les appelants fédérés (SAML et OIDC) ne sont pas AWS des principaux et ne sont pas mis en correspondance par cet `Principal` élément. Le refus ne s'applique donc pas à eux. Cette approche évite d'avoir à recourir à des `Bool` conditions `Null` ou à des conditions complexes pour créer des exceptions pour les appelants fédérés.

> **Note**
>
> La première déclaration autorise toutes les actions pour tous les directeurs. Le refus figurant dans la deuxième instruction est prioritaire pour les AWS mandants authentifiés extérieurs à votre organisation. Les appelants fédérés sont autorisés par la première instruction et ne sont pas affectés par le refus car l'`Principal`élément de l'instruction de refus ne leur correspond pas.

Utiliser les clés de condition VPC pour contrôler l'accès fédéré

Les utilisateurs fédérés peuvent assumer des rôles en utilisant `AssumeRoleWithSAML` ou `AssumeRoleWithWebIdentity` via un point de terminaison VPC. Dans ce cas, vous pouvez utiliser des clés de VPC-specific condition dans la politique de confiance du rôle ou dans les politiques de contrôle des ressources (RCP) pour limiter l'origine de ces demandes. Cela fournit une limite au niveau du réseau en plus des contrôles au niveau de l'identité dans la politique de confiance.

Rubriques

- Comment fonctionnent les clés de condition VPC pour les demandes fédérées
- Exemple : Restreindre la fédération OIDC à un VPC spécifique
- Exemple : restreindre la fédération SAML à un point de terminaison VPC spécifique
- Exemple : Restreindre la fédération OIDC par plage d'adresses IP privées au sein d'un VPC
- Exemple : Restreindre la fédération SAML à plusieurs VPC
- Exemple : combinaison de conditions spécifiques au VPC et au fournisseur
- Exemple : utiliser un RCP pour restreindre l'accès fédéré à un VPC spécifique

Comment fonctionnent les clés de condition VPC pour les demandes fédérées

Les clés de condition disponibles dans le contexte de la demande dépendent du chemin réseau de la demande fédérée.

Disponibilité de la clé de condition VPC par chemin réseau

Clé de condition	Par le biais d'un point de terminaison VPC	Sur Internet public	Description
`aws:SourceVpc`	Oui	Non	ID VPC que traverse la demande
`aws:SourceVpcArn`	Oui	Non	ARN du VPC que traverse la demande
`aws:SourceVpce`	Oui	Non	ID de point de terminaison VPC traversé par la demande
`aws:VpcSourceIp`	Oui	Non	Adresse IP privée de l'appelant au sein du VPC
`aws:SourceIp`	Non	Oui	Adresse IP publique de l'appelant

Si votre politique de confiance utilise `aws:SourceVpc` ou `aws:SourceVpcArn` figure `aws:SourceVpce` dans une `Allow` instruction et que la demande ne passe pas par un point de terminaison VPC, la condition ne correspond pas et refuse AWS implicitement la demande. Cela nécessite en fait que des demandes fédérées traversent le point de terminaison du VPC pour réussir.

> **Important**
>
> Pour les demandes effectuées via un point de terminaison VPC, n'`aws:SourceIp`est pas renseigné. Utilisez-le plutôt `aws:VpcSourceIp` pour IP-based les restrictions.

Nous vous recommandons d'utiliser des politiques `aws:SourceVpcArn` plutôt que `aws:SourceVpc` des politiques de confiance. Les identifiants VPC sont uniques au niveau régional mais pas uniques

au niveau mondial. Ils `aws:SourceVpcArn` incluent la région et le compte, fournissant ainsi une identification unique du VPC au niveau mondial.

> **Note**
>
> Dans les exemples OIDC suivants, la valeur audience (`aud`) utilise`sts.amazonaws.com`, qui est la valeur par défaut pour Amazon EKS. Pour les autres fournisseurs OIDC, remplacez-la par la valeur d'audience configurée pour votre application (par exemple, un ID client ou un URI d'application).

Exemple : Restreindre la fédération OIDC à un VPC spécifique

La politique de confiance suivante permet à un fournisseur OIDC d'assumer le rôle uniquement lorsque la demande provient d'un VPC spécifique via un point de terminaison VPC.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOIDCFromVpc",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "idp.example.com:aud": "sts.amazonaws.com",
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                }
            }
        }
    ]
}
```

AWS refuse implicitement les demandes provenant de l'extérieur du VPC spécifié car `aws:SourceVpcArn` la condition ne correspond pas. Aucune déclaration de refus explicite n'est nécessaire.

Exemple : restreindre la fédération SAML à un point de terminaison VPC spécifique

La politique de confiance suivante permet à un fournisseur SAML d'assumer le rôle uniquement lorsque la demande passe par un point de terminaison VPC spécifique.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSAMLFromVpce",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:saml-provider/ExampleProvider"
            },
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "SAML:aud": "https://signin.aws.amazon.com/saml",
                    "aws:SourceVpce": "vpce-0abcdef1234567890"
                }
            }
        }
    ]
}
```

L'utilisation `aws:SourceVpce` au lieu de `aws:SourceVpc` permet un contrôle plus précis lorsque vous avez plusieurs points de terminaison VPC dans le même VPC et que vous souhaitez restreindre l'accès à un point de terminaison spécifique.

Exemple : Restreindre la fédération OIDC par plage d'adresses IP privées au sein d'un VPC

La politique de confiance suivante limite la fédération aux demandes provenant d'un sous-réseau spécifique au sein du VPC. `aws:VpcSourceIp`À utiliser pour les IP-based restrictions sur les demandes effectuées via un point de terminaison VPC.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOIDCFromSubnet",
```

```
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "idp.example.com:aud": "sts.amazonaws.com",
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                },
                "IpAddress": {
                    "aws:VpcSourceIp": "10.0.1.0/24"
                }
            }
        }
    ]
}
```

> **Important**
>
> Ne pas utiliser `aws:SourceIp` avec des plages d'adresses IP privées pour les demandes via un point de terminaison VPC. La `aws:SourceIp` clé n'est pas disponible pour les demandes de point de terminaison VPC. Utilisez `aws:VpcSourceIp` à la place.

Exemple : Restreindre la fédération SAML à plusieurs VPC

La politique de confiance suivante permet à un fournisseur SAML d'assumer le rôle depuis n'importe lequel des nombreux VPC. Cela est utile lorsque votre organisation possède plusieurs VPC dans différents environnements ou régions qui nécessitent un accès fédéré au même rôle.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSAMLFromMultipleVpcs",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:saml-provider/ExampleProvider"
            },
```

```
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "SAML:aud": "https://signin.aws.amazon.com/saml",
                    "aws:SourceVpcArn": [
                        "arn:aws:ec2:us-west-2:111122223333:vpc/vpc-111aaa11",
                        "arn:aws:ec2:us-east-1:111122223333:vpc/vpc-222bbb22"
                    ]
                }
            }
        }
    ]
}
```

Exemple : combinaison de conditions spécifiques au VPC et au fournisseur

Vous pouvez combiner les clés de condition VPC avec des conditions spécifiques au fournisseur, telles que l'audience (`aud`) et le sujet (`sub`) pour un contrôle d'accès précis. La politique de confiance suivante limite l'attribution des rôles à un utilisateur fédéré spécifique et exige que la demande provienne d'un VPC spécifique.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOIDCUserFromVpc",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "idp.example.com:aud": "sts.amazonaws.com",
                    "idp.example.com:sub": "user123",
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                }
            }
        }
    ]
}
```

Cela combine des contrôles au niveau de l'identité (limités à un utilisateur fédéré spécifique) avec des contrôles au niveau du réseau (limités à un VPC spécifique), offrant ainsi une défense approfondie.

Exemple : utiliser un RCP pour restreindre l'accès fédéré à un VPC spécifique

La politique de contrôle des ressources (RCP) suivante refuse `AssumeRoleWithSAML` et `AssumeRoleWithWebIdentity` demande, sauf si elles proviennent d'un VPC spécifique. Contrairement aux politiques de confiance qui s'appliquent par rôle, une politique de contrôle des ressources (RCP) s'applique à tous les rôles des comptes cibles. Cela fournit une limite de réseau centralisée.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyFederatedAccessOutsideVpc",
            "Effect": "Deny",
            "Principal": "*",
            "Action": [
                "sts:AssumeRoleWithWebIdentity",
                "sts:AssumeRoleWithSAML"
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                },
                "Null": {
                    "aws:SourceVpcArn": "false"
                }
            }
        }
    ]
}
```

Cette `Null` condition garantit que le refus ne s'applique que s'il `aws:SourceVpcArn` est présent dans le contexte de la demande (c'est-à-dire si la demande est passée par un point de terminaison VPC mais provient du mauvais VPC). Sans cela, les demandes sur l'Internet public (où il `aws:SourceVpcArn` n'y en a pas) seraient également refusées par`StringNotEquals`. Si vous supprimez la `Null` vérification, le refus s'applique également aux demandes qui ne transitent

par aucun point de terminaison VPC, y compris les demandes via Internet public. Nous vous recommandons de procéder à des tests approfondis avant de supprimer l'`Null`enregistrement dans les environnements de production.

AWS services qui fonctionnent avec IAM

Les AWS services répertoriés ci-dessous sont regroupés par ordre alphabétique et incluent des informations sur les fonctionnalités IAM qu'ils prennent en charge :

- Service : vous pouvez choisir le nom d'un service pour consulter la AWS documentation relative à l'autorisation IAM et à l'accès à ce service.
- Actions : Vous pouvez spécifier des actions individuelles dans une politique. Si le service ne prend pas en charge cette fonction, Toutes les actions est sélectionné dans l'éditeur visuel. Dans un document de politique JSON, vous devez utiliser le caractère générique `*` dans l'élément `Action`. Pour obtenir la liste des actions de chaque service, consultez la section Actions, ressources et clés de condition pour les AWS services.
- Resource-level autorisations — Vous pouvez utiliser les ARN pour spécifier des ressources individuelles dans la politique. Si le service ne prend pas en charge cette fonction, Toutes les ressources est sélectionné dans l'éditeur visuel de politique. Dans un document de politique JSON, vous devez utiliser le caractère générique `*` dans l'élément `Resource`. Certaines actions, comme les actions `List*`, ne prennent pas en charge la spécification d'un ARN, car elles sont conçues pour renvoyer plusieurs ressources. Si un service prend en charge cette fonctionnalité pour certaines ressources, mais pas pour d'autres, cela est indiqué par Partiel dans le tableau. Consultez la documentation de ce service pour plus d'informations.
- Resource-based politiques : vous pouvez associer des politiques basées sur les ressources à une ressource au sein du service. Resource-based les politiques incluent un `Principal` élément permettant de spécifier quelles identités IAM peuvent accéder à cette ressource. Pour plus d'informations, consultez Identity-based politiques et politiques basées sur les ressources.
- ABAC (Autorisation basée sur les balises) : pour contrôler l'accès basé sur des balises, vous devez fournir les informations de balises dans l'élément de condition d'une politique utilisant les clés de condition `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* ou `aws:TagKeys`. Si un service prend en charge les trois clés de condition pour tous les types de ressources, alors la valeur pour ce service est Oui. Si un service prend en charge les trois clés de condition pour certains types de ressources uniquement, la valeur est Partielle.

Pour de plus amples informations sur la définition d'autorisations basées sur des attributs tels que des balises, veuillez consulter Définition des autorisations en fonction des attributs avec autorisation ABAC. Pour accéder à un didacticiel décrivant les étapes de configuration de l'ABAC, consultez Use attribute-based access control (ABAC). (Utilisation du contrôle d'accès basé sur les attributs).

- Informations d'identification temporaires : vous pouvez utiliser les informations d'identification à court terme que vous obtenez lorsque vous vous connectez à l'aide d'IAM Identity Center, du gestionnaire d'accès au compte, lorsque vous changez de rôle dans la console ou que vous générez AWS STS à l'aide de l' AWS API AWS CLI or. Vous pouvez accéder aux services avec une valeur No uniquement en utilisant vos informations d'identification utilisateur IAM à long terme. Cela inclut un nom d'utilisateur et un mot de passe ou vos clés d'accès utilisateur. Pour plus d'informations, consultez Informations d'identification de sécurité temporaires dans IAM.
- Service-linked rôles — Un rôle lié à un service est un type spécial de rôle de service qui autorise le service à accéder aux ressources d'autres services en votre nom. Cliquez sur le lien Oui ou Partiel pour afficher la documentation pour les services qui prennent en charge ces rôles. Cette colonne n'indique pas si le service utilise des rôles de service standard. Pour plus d'informations, consultez la section Service-linked Rôles.
- Autres informations : Si un service ne prend pas entièrement en charge une fonction, vous pouvez consulter les notes de bas de page d'une entrée pour afficher les limitations et les liens vers les informations connexes.

Services qui fonctionnent avec IAM

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Gestionnaire d'accès au compte	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	Oui
Gestion de compte AWS	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS Recommandations d'action	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Activate Console	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Opération d'IA Amazon	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Amplify Admin	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Amplify	✓ Oui	✓ Oui	✗ Non	⚠ Partielle	✓ Oui	✗ Non
AWS Amplify Générateur d'interface utilisateur	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
API Apache Kafka pour les clusters Amazon MSK	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon API Gateway	✓ Oui	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui
Gestion Amazon API Gateway	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Gestion Amazon API Gateway V2	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Studio d'applications	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS App2Container	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS AppConfig	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS AppFabric	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Amazon AppFlow	Oui	Oui	Non	Oui	Oui	Non
Amazon AppIntegrations	Oui	Oui	Non	Oui	Oui	Oui
Application Autoscaling	Oui	Oui	Non	Oui	Oui	Oui
AWS Arsenal de découverte d'applications	Oui	Non	Non	Non	Oui	Non
AWS Application Discovery Service	Oui	Non	Non	Non	Oui	Oui
AWS Transform MGN	Oui	Oui	Non	Oui	Oui	Oui
Amazon Application Recovery Controller – Changement de zone	Oui	Oui	Non	Non	Oui	Non
AWS Service de transformation des applications	Oui	Non	Non	Non	Oui	Non
AWS App Mesh	Oui	Oui	Non	Oui	Oui	Oui
AWS App Mesh Version préliminaire	Oui	Oui	Non	Non	Oui	Oui
AWS App Runner	Oui	Oui	Non	Oui	Oui	Oui
Amazon AppStream 2.0	Oui	Oui	Non	Oui	Oui	Non
AWS AppSync	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Commutateur de région Amazon ARC	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Artifact	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Athena	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Audit Manager	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon Aurora DSQL	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Auto Scaling	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✓ Oui
AWS Échange de données B2B	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Backup	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✓ Oui
AWS Backup Passerelle	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Backup Rechercher	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Backup rangement	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Batch	✓ Oui	⚠ Partielle	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon Bedrock	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon Bedrock AgentCore	✓ Oui	✓ Oui	✗ Non	⚠ Partielle	✓ Oui	✗ Non
AWS Billing and Cost Management	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Tableaux de bord de Facturation et gestion des coûts AWS	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Billing and Cost Management Exportations de données	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Billing and Cost Management Calculateur de prix	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Billing and Cost Management Actions recommandées	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Billing Conductor	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon Braket	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Service du budget	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Certificate Manager (ACM)	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon Q Developer dans les applications de chat	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon Chime	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Clean Rooms	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Clean Rooms ML	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Client VPN	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✓ Oui

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS Cloud9	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✓ Oui
AWS Cloud API de contrôle	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Cloud Directory	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
CloudFormation	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon CloudFront	✓ Oui	✓ Oui	✗ Non	⚠ Partielle	✓ Oui	✓ Oui
Amazon CloudFront KeyValueStore	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS CloudHSM	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Cloud Map	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon CloudSearch	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
AWS CloudShell	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
AWS CloudTrail	✓ Oui	✓ Oui	⚠ Partiel (Informations)	✓ Oui	✓ Oui	✓ Oui
AWS CloudTrail Données	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon CloudWatch	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	⚠ Partiel (Informations)

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Informations sur les CloudWatch applications Amazon	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Signaux CloudWatch d'application Amazon	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon CloudWatch Internet Monitor	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon CloudWatch Logs	✓ Oui	✓ Oui	✓ Oui	⚠ Partielle	✓ Oui	✓ Oui
Moniteur synthétique Amazon CloudWatch Network	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Gestionnaire d'accès Amazon CloudWatch Observability	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Service d'administration Amazon CloudWatch Observability	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon CloudWatch RUM	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon CloudWatch Synthetics	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS CodeArtifact	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✗ Non
AWS CodeBuild	✓ Oui	✓ Oui	✓ Oui (Informations)	⚠ Partiel (Informations)	✓ Oui	✗ Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Amazon CodeCatalyst	Oui	Oui	Non	Oui	Oui	Oui
AWS CodeCommit	Oui	Oui	Non	Oui	Oui	Non
AWS CodeConnections	Oui	Oui	Non	Oui	Oui	Non
AWS CodeDeploy	Oui	Oui	Non	Oui	Oui	Non
AWS CodeDeploy service de commandes hôte sécurisé	Oui	Non	Non	Non	Oui	Non
Amazon CodeGuru Profiler	Oui	Oui	Non	Oui	Oui	Oui
CodeGuru Réviseur Amazon	Oui	Oui	Non	Oui	Oui	Oui
Amazon CodeGuru Security	Oui	Oui	Non	Oui	Oui	Non
AWS CodePipeline	Oui	Partielle	Non	Oui	Oui	Non
AWS CodeStar	Oui	Partielle	Non	Oui	Oui	Non
AWS CodeStar Connexions	Oui	Oui	Non	Oui	Oui	Oui
AWS CodeStar Notifications	Oui	Oui	Non	Oui	Oui	Oui
Amazon CodeWhisperer	Oui	Oui	Non	Oui	Oui	Oui
Amazon Cognito	Oui	Oui	Non	Oui	Oui	Oui
Amazon Cognito Sync	Oui	Oui	Non	Non	Oui	Oui

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Groupes d'utilisateurs Amazon Cognito	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon Comprehend	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon Comprehend Medical	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Optimiseur de calcul AWS	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✓ Oui
AWS Config	✓ Oui	⚠ Partiel (Informations)	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon Connect Customer	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Cas clients d'Amazon Connect	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Profils des clients Amazon Connect	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Campagnes sortantes destinées aux clients Amazon Connect	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Identifiant vocal du client Amazon Connect	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon Connect Health	✓ Oui	✓ Oui	✗ Non	⚠ Partielle	✓ Oui	✗ Non
AWS Console Mobile Application	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS Facturation consolidée	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Control Catalog	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Control Tower	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Cost and Usage Report	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Cost Explorer	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Hub d'optimisation des coûts AWS	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Service de vérification des clients AWS	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Database Migration Service	✓ Oui	✓ Oui	✗ Non (Informations)	✓ Oui	✓ Oui	✓ Oui
Database Query Metadata Service	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Data Exchange	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon Data Lifecycle Manager	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Data Pipeline	✓ Oui	✓ Oui	✗ Non	⚠ Partielle	✓ Oui	✗ Non
AWS DataSync	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon DataZone	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS Deadline Cloud	Oui	Oui	Non	Oui	Oui	Non
AWS DeepRacer	Oui	Oui	Non	Oui	Oui	Oui
Amazon Detective	Oui	Oui	Non	Oui	Oui	Non
AWS Device Farm	Oui	Oui	Non	Oui	Oui	Oui
Amazon DevOps Guru	Oui	Oui	Non	Non	Oui	Oui
Outils de diagnostic AWS	Oui	Oui	Non	Oui	Oui	Non
AWS Direct Connect	Oui	Oui	Non	Oui	Oui	Oui
AWS Directory Service	Oui	Oui	Non	Oui	Oui	Oui
AWS Données du service de répertoire	Oui	Oui	Non	Oui	Oui	Non
Clusters élastiques Amazon DocumentDB	Oui	Oui	Non	Oui	Oui	Oui
Amazon DynamoDB Accelerator (DAX)	Oui	Oui	Non	Non	Oui	Oui
Amazon DynamoDB	Oui	Oui	Oui	Oui	Oui	Oui
Amazon Elastic Compute Cloud (Amazon EC2)	Oui	Partielle	Non	Oui	Oui	Partiel (Informations)
Amazon EC2 Auto Scaling	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
EC2 Image Builder	Oui	Oui	Non	Oui	Oui	Oui
Amazon EC2 Instance Connect	Oui	Oui	Non	Oui	Oui	Oui
AWS Elastic Beanstalk	Oui	Partielle	Non	Oui	Oui	Oui
Amazon Elastic Block Store (Amazon EBS)	Oui	Partielle	Non	Oui	Oui	Non
Amazon Elastic Container Registry (Amazon ECR)	Oui	Oui	Oui	Oui	Oui	Oui
Amazon Elastic Container Registry Public (Amazon ECR Public)	Oui	Oui	Non	Oui	Oui	Non
Amazon Elastic Container Service (Amazon ECS)	Oui	Partiel (Informations)	Non	Oui	Oui	Oui
Reprise après sinistre AWS Elastic	Oui	Oui	Non	Oui	Oui	Oui
Amazon Elastic File System (Amazon EFS)	Oui	Oui	Oui	Partielle	Oui	Oui
Amazon Elastic Kubernetes Service (Amazon EKS)	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Authentification Amazon Elastic Kubernetes Service (Amazon EKS)	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Elastic Load Balancing	✓ Oui	⚠ Partielle	✗ Non	⚠ Partielle	✓ Oui	✓ Oui
Service Amazon Elastic VMware	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon ElastiCache	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Service d'activation des appliances élémentaires et des logiciels	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Appliances et logiciels Elemental	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Elemental MediaConnect	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✓ Oui
AWS Elemental MediaConvert	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Elemental MediaLive	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Elemental MediaPackage	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	⚠ Partiel (Informations)
AWS Elemental MediaPackage V2	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS Elemental MediaPackage VOD	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	⚠ Partiel (Informations)
AWS Elemental MediaTailor	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Dossiers de support élémentaires	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Contenu de support élémentaire	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon EMR	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon EMR on EKS	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon EMR sans serveur	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS End User Messaging SMS et Voice V2	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Messagerie sociale destinée aux utilisateurs finaux	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Résolution des entités AWS	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✗ Non
Amazon EventBridge	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✗ Non
Amazon EventBridge Pipes	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon EventBridge Scheduler	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
EventBridge Schémas Amazon	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✗ Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS Service d'injection de défauts	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon FinSpace	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon FinSpace API	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Firewall Manager	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	⚠ Partielle
Fleet Hub for AWS IoT Device Management	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon Forecast	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon Fraud Detector	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
FreeRTOS	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Niveau gratuit	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon FSx	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
GameLift Serveurs Amazon	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon GameLift Streams	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Global Accelerator	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Glue	✓ Oui	✓ Oui	⚠ Partielle	⚠ Partielle	✓ Oui	✗ Non
AWS Glue DataBrew	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Ground Station	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Étiquetage Amazon Ground Truth	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon GuardDuty	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Health API et notifications	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
AWS HealthImaging	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS HealthLake	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS HealthOmics	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS IAM Identity Center	✓ Oui	✓ Oui	✗ Non	⚠ Partielle	✓ Oui	✓ Oui
Répertoire d'IAM Identity Center	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Magasin d'identités d'IAM Identity Center	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Service OIDC d'IAM Identity Center	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Gestion des identités et des accès AWS (JE SUIS)	✓ Oui	✓ Oui	⚠ Partiel (Informations)	⚠ Partiel (Informations)	⚠ Partiel (Informations)	✗ Non
Gestion des identités et des accès AWS et Access Analyzer	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	⚠ Partielle

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Gestion des identités et des accès AWS Des rôles n'importe où	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Authentification du magasin d'identités	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Synchronisation des identités	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Import/Export	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Inspector	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon Inspector Classic	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✓ Oui
Amazon InspectorScan	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Interactive Video Service	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Service de vidéo interactive Amazon Chat	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Facturation AWS	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS IoT Analytics	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS IoT	✓ Oui	✓ Oui	⚠ Partiel (Informations)	✓ Oui	✓ Oui	✗ Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS IoT Core Conseiller en matière d'appareils	Oui	Oui	Non	Oui	Oui	Non
AWS IoT Testeur d'appareils	Oui	Non	Non	Non	Oui	Non
AWS IoT Events	Oui	Oui	Non	Oui	Oui	Non
AWS IoT FleetWise	Oui	Oui	Non	Oui	Oui	Non
AWS IoT Greengrass	Oui	Oui	Non	Oui	Oui	Non
AWS IoT Greengrass V2	Oui	Oui	Non	Partielle	Oui	Non
AWS IoT Tâches DataPlane	Oui	Oui	Non	Non	Oui	Non
AWS IoT Service d'intégrations gérées	Oui	Oui	Non	Oui	Oui	Oui
AWS IoT SiteWise	Oui	Oui	Non	Oui	Oui	Oui
AWS IoT TwinMaker	Oui	Oui	Non	Oui	Oui	Oui
AWS IoT Wireless	Oui	Oui	Non	Oui	Oui	Non
Amazon Kendra	Oui	Oui	Non	Oui	Oui	Non
Amazon Kendra Intelligent Ranking (Classement intelligent Amazon Kendra)	Oui	Oui	Non	Oui	Oui	Non
AWS Key Management Service (AWS KMS)	Oui	Oui	Oui	Oui	Oui	Oui

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Amazon Keyspaces (pour Apache Cassandra)	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Service géré Amazon pour Apache Flink	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Service géré Amazon pour Apache Flink V2	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon Data Firehose	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon Kinesis Data Streams	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✗ Non
Amazon Kinesis Video Streams	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Lake Formation	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✓ Oui
AWS Lambda	✓ Oui	✓ Oui	✓ Oui	⚠ Partiel (Informations)	✓ Oui	⚠ Partiel (Informations)
AWS Launch Wizard	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Lex	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon Lex V2	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✗ Non	✓ Oui
AWS License Manager	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS License Manager Gestionnaire des abonnements Linux	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS License Manager Abonnements d'utilisateurs	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✓ Oui
Amazon Lightsail	✓ Oui	⚠ Partiel (Informations)	✗ Non	⚠ Partiel (Informations)	✓ Oui	✓ Oui
Amazon Location Service	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Cartes Amazon Location Service	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Emplacements Amazon Location Service	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Routes Amazon Location Service	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Lookout for Equipment	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon Lookout for Metrics	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon Machine Learning	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Macie	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Mainframe Modernization	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Mainframe Modernization Tests d'applications	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon Managed Blockchain	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Amazon Managed Blockchain Query	Oui	Non	Non	Non	Oui	Non
Amazon Managed Grafana	Oui	Oui	Non	Oui	Oui	Oui
Amazon Managed Service for Prometheus	Oui	Oui	Non	Oui	Oui	Non
Amazon Managed Streaming for Apache Kafka (MSK)	Oui	Oui	⚠ Partiel (Informations)	Oui	Oui	Oui
Amazon Managed Streaming for Kafka Connect	Oui	Oui	Non	Oui	Oui	Oui
Amazon Managed Workflows for Apache Airflow	Oui	Oui	Non	Oui	Oui	Oui
AWS Marketplace	Oui	Non	Non	Non	Oui	Oui
AWS Marketplace Catalogue	Oui	Oui	Non	Oui	Oui	Non
AWS Marketplace Commerce Analytics	Oui	Non	Non	Non	Non	Non
Service de déploiement AWS Marketplace	Oui	Oui	Non	Oui	Oui	Non
AWS Marketplace Découverte	Oui	Non	Non	Non	Oui	Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS Marketplace Entitlement Service	Oui	Non	Non	Non	Oui	Non
AWS Marketplace Service de création d'images	Oui	Non	Non	Non	Oui	Non
Portail de gestion AWS Marketplace	Oui	Non	Non	Non	Oui	Non
AWS Marketplace Metering Service	Oui	Non	Non	Non	Oui	Non
AWS Marketplace Marketplace privée	Oui	Non	Non	Non	Oui	Non
AWS Marketplace Intégration des systèmes d'approvisionnement	Oui	Non	Non	Non	Oui	Non
AWS Marketplace Génération de rapports	Oui	Oui	Non	Non	Oui	Non
AWS Marketplace Rapports sur les vendeurs	Oui	Oui	Non	Non	Oui	Non
AWS Marketplace Informations sur les fournisseurs	Oui	Oui	Non	Oui	Oui	Non
Amazon MediaImport	Oui	Non	Non	Non	Non	Non
Amazon MemoryDB	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Amazon Message Delivery Service	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Message Gateway Service	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Microservice Extractor for .NET	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Crédits du programme Migration Acceleration	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Migration Hub	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✓ Oui
Orchestrateur AWS Migration Hub	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Migration Hub Refactor Spaces	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✓ Oui
Recommandations de stratégie AWS Migration Hub	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✓ Oui
Amazon Monitron	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon MQ	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon Neptune	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✓ Oui

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Multi-partyapprobation	Oui	Oui	Non	Partiel (Informations)	Oui	Non
Amazon Neptune Analytics	Oui	Oui	Non	Oui	Oui	Non
AWS Network Firewall	Oui	Oui	Non	Oui	Oui	Oui
Surveillance du flux réseau	Oui	Oui	Non	Oui	Oui	Non
AWS Network Manager	Oui	Oui	Non	Oui	Oui	Oui (Informations)
AWS Network Manager Chat	Oui	Non	Non	Non	Oui	Non
Amazon Nimble Studio	Oui	Oui	Non	Oui	Oui	Non
Amazon One Enterprise	Oui	Oui	Non	Oui	Oui	Non
Amazon OpenSearch	Oui	Oui	Non	Non	Oui	Non
OpenSearchIngestion d'Amazon	Oui	Oui	Non	Oui	Oui	Oui
Amazon OpenSearch sans serveur	Oui	Oui	Non	Oui	Oui	Oui
Amazon OpenSearch Service	Oui	Oui	Oui	Oui	Oui	Oui
Base de données Oracle@AWS	Oui	Oui	Non	Oui	Oui	Non
AWS OpsWorks	Oui	Oui	Non	Non	Oui	Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS OpsWorks Gestion de configuration	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Organizations	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✗ Non	✓ Oui
AWS Outposts	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Parallel Computing Service	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Gestion des comptes Partner Central	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Partenaire : Central Selling	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Payment Cryptography	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Paiements	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Performance Insights	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon Personalize	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Pinpoint	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Service de messagerie Amazon Pinpoint	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Service de messages SMS et vocaux Amazon Pinpoint	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Polly	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS Price List	Oui	Non	Non	Non	Oui	Non
AWS 5G privée	Oui	Oui	Non	Oui	Oui	Non
CA privée AWS Connector for Active Directory (français non garanti)	Oui	Oui	Non	Oui	Oui	Non
Connecteur CA privée AWS for SCEP	Oui	Oui	Non	Oui	Oui	Non
Autorité de certification privée AWS (CA privée AWS)	Oui	Oui	Oui	Oui	Oui	Non
AWS PrivateLink	Oui	Non	Non	Non	Oui	Non
AWS Proton	Oui	Oui	Non	Oui	Oui	Oui
AWS Console de bons de commande	Oui	Oui	Non	Oui	Oui	Non
Amazon Q Business	Oui	Oui	Non	Oui	Oui	Oui
Applications Amazon Q Business Q	Oui	Oui	Non	Oui	Oui	Oui
Amazon Q Developer	Oui	Oui	Non	Oui	Oui	Oui
Amazon Q in Connect	Oui	Oui	Non	Oui	Oui	Non
Amazon Quantum Ledger Database (Amazon QLDB)	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Amazon Quick	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon RDS Data API	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Authentification IAM Amazon RDS	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Corbeille	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon Redshift	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
API de données Amazon Redshift	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Redshift sans serveur	✓ Oui	✓ Oui	✓ Oui (Informations)	✓ Oui	✓ Oui	✗ Non
Amazon Rekognition	✓ Oui	✓ Oui	⚠ Partiel (Informations)	✓ Oui	✓ Oui	✗ Non
Amazon Relational Database Service (Amazon RDS) (Informations)	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS re:Post Privé	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Resilience Hub	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS Resource Access Manager (AWS RAM)	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Explorateur de ressources AWS	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Groupes de ressources AWS	✓ Oui	✓ Oui	✗ Non	✓ Oui	⚠ Partiel (Informations)	✓ Oui
AWS Resource Groups Tagging API	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Portail Amazon RHEL Knowledgebase	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS RoboMaker	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon Route 53	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Route 53 Domaines	✓ Oui	✗ Non	✗ Non	✗ Non	✗ Non	✗ Non
Profils Amazon Route 53	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon Route 53 Recovery Cluster	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Configuration d'Amazon Route 53 Recovery Controls	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon Route 53 Recovery Readiness	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Amazon Route 53 Resolver	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Tissu RTB	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon S3 Express	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✗ Non
Amazon Glacier	✓ Oui	✓ Oui	✓ Oui	✗ Non	✓ Oui	⚠ Partielle
Tables Amazon S3	✓ Oui	✓ Oui	✓ Oui	✗ Non	✓ Oui	✗ Non
Amazon S3 Vectors	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon SageMaker AI	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	⚠ Partiel (Informations)
Assistant de science des données Amazon SageMaker AI	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Fonctionnalités géospatiales d'Amazon SageMaker AI	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
Amazon SageMaker AI avec MLflow	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Savings Plans AWS	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Secrets Manager	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✗ Non
AWS Security Hub CSPM	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Réponse aux incidents de sécuritéAWS	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Amazon Security Lake	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✓ Oui
AWS Security Token Service (AWS STS)	✓ Oui	⚠ Partiel (Informations)	✗ Non	✓ Oui	⚠ Partiel (Informations)	✗ Non
AWS Serverless Application Repository	✓ Oui	✓ Oui	✓ Oui	✗ Non	✓ Oui	✗ Non
AWS Service Catalog	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Service de gestion des fonctionnalités de l'expérience utilisateur de la AWS console	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Service Quotas	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Shield	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Contrôleur de sécurité réseau AWS Shield	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Signer	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Sign-In	✓ Oui	✗ Non	✓ Oui	✗ Non	✓ Oui	✗ Non
Amazon SimpleDB	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Simple Email Service – Gestionnaire de messagerie	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
Amazon Simple Email Service (Amazon SES) v2	✓ Oui	⚠ Partiel (Informations)	✓ Oui	✓ Oui	⚠ Partiel (Informations)	✓ Oui
Amazon Simple Notification Service (Amazon SNS)	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✓ Oui	✗ Non
Amazon Simple Queue Service (Amazon SQS)	✓ Oui	✓ Oui	✓ Oui	✓ Partielle	✓ Oui	✗ Non
Amazon Simple Storage Service (Amazon S3)	✓ Oui	✓ Oui	✓ Oui	⚠ Partiel (Informations)	✓ Oui	⚠ Partiel (Informations)
Amazon Simple Storage Service(Amazon S3) Objet Lambda	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Simple Storage Service (Amazon S3) sur AWS Outposts	✓ Oui	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui
Amazon Simple Workflow Service (Amazon SWF)	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS SimSpaceTisserand	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Site-to-Site VPN	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✓ Oui
AWS Snowball Edge	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS Snowball Edge Edge	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Snowball Edge Device Management	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS SQL Workbench	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Step Functions	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Storage Gateway	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Supply Chain	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Support	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✓ Oui
AWS Support App in Slack	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Support Console	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Support Plans	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Durabilité	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
AWS Systems Manager	✓ Oui	✓ Oui	⚠ Partielle	✓ Oui	✓ Oui	✓ Oui
AWS Systems Manager pour SAP	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Systems Manager GUI Connect	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS Systems Manager Incident Manager	Oui	Oui	Oui	Oui	Oui	Oui
AWS Systems Manager Incident Manager Contacts	Oui	Oui	Oui	Non	Oui	Non
AWS Systems Manager Configuration rapide	Oui	Oui	Non	Oui	Oui	Non
Tag Editor	Oui	Non	Non	Non	Oui	Non
AWS Paramètres fiscaux	Oui	Non	Non	Non	Oui	Non
AWS Générateur de réseaux de télécommunications	Oui	Oui	Non	Oui	Oui	Non
Amazon Textract	Oui	Non	Non	Non	Oui	Non
Amazon Timestream	Oui	Oui	Non	Oui	Oui	Non
Amazon Timestream Influxdb	Oui	Oui	Non	Oui	Oui	Oui
AWS API Tiros (pour Reachability Analyzer)	Oui	Non	Non	Non	Non	Non
Amazon Transcribe	Oui	Oui	Non	Oui	Oui	Non
AWS Transfer Family	Oui	Oui	Non	Oui	Oui	Non
AWS Transform	Oui	Oui	Non	Non	Oui	Non
Amazon Translate	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS Trusted Advisor	⚠ Partiel (Informations)	✓ Oui	✗ Non	✗ Non	⚠ Partielle	✓ Oui
AWS Notifications aux utilisateurs	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS Contacts pour les notifications des utilisateurs	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✗ Non
AWS Abonnements d'utilisateurs	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Accès vérifié par AWS	✓ Oui	✗ Non	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Verified Permissions	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
Amazon Virtual Private Cloud (Amazon VPC)	✓ Oui	⚠ Partiel (Informations)	⚠ Partiel (Informations)	✓ Oui	✓ Oui	⚠ Partiel (Informations)
Amazon VPC Lattice	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Amazon VPC Lattice Services	✓ Oui	✓ Oui	✗ Non	✗ Non	✓ Oui	✗ Non
AWS WAF	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
Classique AWS WAF	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui
AWS WAF Régional	✓ Oui	✓ Oui	✗ Non	✓ Oui	✓ Oui	✓ Oui

Service	Actions	Resource-level permissions	Resource-based politiques	ABAC	Informations d'identification temporaires	Service-linked rôles
AWS Well-Architected Tool	Oui	Oui	Non	Oui	Oui	Non
AWS Wickr	Oui	Oui	Non	Oui	Oui	Non
Amazon WorkMail	Oui	Oui	Non	Oui	Oui	Oui
Flux de WorkMail messages Amazon	Oui	Oui	Non	Non	Oui	Non
Amazon WorkSpaces	Oui	Oui	Non	Oui	Oui	Non
AWS WorkSpaces Instances gérées	Oui	Oui	Non	Oui	Oui	Non
Navigateur Amazon WorkSpaces Secure	Oui	Oui	Non	Oui	Oui	Oui
Amazon WorkSpaces Thin Client	Oui	Oui	Non	Oui	Oui	Non
AWS X-Ray	Oui	Partiel (Informations)	Non	Partiel (Informations)	Oui	Non

En savoir plus

AWS CloudTrail

CloudTrail prend en charge les politiques basées sur les ressources sur les magasins de données d'événements, les tableaux de bord et les canaux de CloudTrail Lake utilisés pour les intégrations avec des sources d'événements extérieures à. AWS

Amazon CloudWatch

CloudWatch les rôles liés à un service ne peuvent pas être créés à l'aide de la fonctionnalité Alarm Actions Console de gestion AWS, et ne prennent en charge que celle-ci.

AWS CodeBuild

CodeBuild prend en charge le partage de ressources entre comptes en utilisant AWS RAM.

CodeBuild soutient ABAC pour les actions basées sur des projets.

AWS Config

AWS Config prend en charge les autorisations au niveau des ressources pour l'agrégation de données multicomptes et multirégions et les règles. AWS Config Pour obtenir la liste des ressources prises en charge, consultez les sections Agrégation de Multi-Account Multi-Region données et AWS Config Règles du guide de l'AWS Config API.

AWS Database Migration Service

Vous pouvez créer et modifier des politiques associées aux clés de AWS KMS chiffrement que vous créez pour chiffrer les données migrées vers les points de terminaison cibles pris en charge. Les points de terminaison cibles pris en charge incluent Amazon Redshift et Amazon S3. Pour plus d'informations, consultez les sections Création et utilisation de AWS KMS clés pour chiffrer les données cibles Amazon Redshift et AWS KMS Création de clés pour chiffrer des objets cibles Amazon S3 AWS Database Migration Service dans le guide de l'utilisateur.

Amazon Elastic Compute Cloud

Les rôles liés au service Amazon EC2 ne peuvent être utilisés que pour les fonctionnalités suivantes : demandes d'instances Spot, demandes de parc d'instances Spot, flotte d'Amazon EC2 et lancement rapide pour les instances Windows.

Amazon Elastic Container Service

Seules quelques actions Amazon ECS prennent en charge les autorisations de niveau ressource.

AWS Elemental MediaPackage

MediaPackage prend en charge les rôles liés aux services pour publier les journaux d'accès des clients CloudWatch , mais pas pour d'autres actions d'API.

Gestion des identités et des accès AWS

IAM prend en charge un seul type de politique basée sur les ressources, nommé politique d'approbation de rôle, qui est attaché à un rôle IAM. Pour plus d'informations, consultez Octroi d'autorisations à un utilisateur pour endosser un rôle.

IAM prend en charge le contrôle d'accès basé sur des balises pour la plupart des ressources IAM. Pour plus d'informations, consultez Tags pour Gestion des identités et des accès AWS resources.

Seules quelques actions d'API pour IAM peuvent être appelées avec des informations d'identification temporaires. Pour plus d'informations, consultez Comparaison de vos options d'API.

AWS IoT

Les appareils connectés AWS IoT sont authentifiés à l'aide de X.509 certificats ou à l'aide d'Amazon Cognito Identities. Vous pouvez associer des AWS IoT politiques à un X.509 certificat ou à Amazon Cognito Identity pour contrôler ce que l'appareil est autorisé à faire. Pour plus d'informations, consultez Sécurité et identité pour AWS IoT dans le Manuel du développeur AWS IoT .

AWS Lambda

Lambda prend en charge le contrôle d'accès par attributs (ABAC) pour les fonctions, les mappages des sources d'événements et les configurations de signature de code. Les couches ne sont pas prises en charge. Pour plus d'informations, consultez Utilisation du contrôle d'accès par attributs dans Lambda.

Lambda n'a pas de rôles liés au service, contrairement à Lambda@Edge. Pour plus d'informations, consultez la section Service-Linked Rôles pour Lambda @Edge dans le manuel Amazon CloudFront Developer Guide.

Amazon Lightsail

Lightsail prend partiellement en charge les autorisations au niveau des ressources et ABAC. Pour de plus amples informations, veuillez consulter la rubrique Actions, ressources et clés de condition pour Amazon Lightsail.

Amazon Managed Streaming for Apache Kafka (MSK)

Vous pouvez associer une politique de cluster à un cluster Amazon MSK configuré pour une connectivité multi-VPC.

Multi-party approbation

Pour les sources d'identité d' Multi-party approbation et les équipes d'approbation, vous pouvez utiliser les trois clés de condition ABAC : `aws:ResourceTag/`*`key-name`*`aws:RequestTag/`*`key-name`*, et`aws:TagKeys`.

Pour les sessions, vous ne pouvez utiliser que la clé de condition `aws:ResourceTag/`*`key-name`*, car :

- Les sessions ne sont pas des ressources directement balisables.
- Les sessions héritent des balises de l'équipe d'approbation qui leur est associée.

AWS Network Manager

AWS Le Cloud WAN prend également en charge les rôles liés aux services. Pour plus d'informations, consultez la section Rôles liés au service AWS Cloud WAN dans le guide Amazon VPC AWS Cloud WAN.

Amazon Redshift sans serveur

Redshift sans serveur prend en charge les politiques basées sur les ressources pour le partage d'instantanés.

Amazon Relational Database Service

Amazon Aurora est un moteur de base de données relationnelle entièrement géré compatible avec MySQL et PostgreSQL. Vous pouvez choisir Aurora MySQL ou Aurora PostgreSQL en tant qu'option de moteur de base de données lorsque vous configurez de nouveaux serveurs de base de données via Amazon RDS. Pour plus d'informations, consultez Gestion des identités et des accès pour Amazon Aurora dans le Manuel de l'utilisateur Amazon Aurora.

Amazon Rekognition

Resource-based les politiques ne sont prises en charge que pour la copie des modèles d'étiquettes personnalisées Amazon Rekognition.

Groupes de ressources AWS

Les utilisateurs peuvent endosser un rôle avec une politique qui autorise les opérations de groupes de ressources.

Amazon SageMaker AI

Service-linked des rôles sont actuellement disponibles pour SageMaker AI Studio et SageMaker AI Training.

AWS Security Token Service

AWS STS ne dispose pas de « ressources », mais permet de restreindre l'accès de la même manière aux utilisateurs. Pour de plus amples informations, veuillez consulter Refus de l'accès aux informations d'identification de sécurité temporaires par nom.

Seules certaines des opérations d'API pour les appels d' AWS STS assistance avec des informations d'identification temporaires. Pour plus d'informations, consultez Comparaison de vos options d'API.

Amazon Simple Email Service

Vous pouvez uniquement utiliser les autorisations au niveau des ressources dans les instructions de politique qui font référence à des actions liées à l'envoi d'e-mails, comme `ses:SendEmail` ou `ses:SendRawEmail`. Pour obtenir des instructions de politique qui font référence à d'autres actions, l'élément Resource peut uniquement contenir `*`.

Seule l'API Amazon SES prend en charge les informations d'identification de sécurité temporaires. L'interface SMTP Amazon SES ne prend pas en charge les informations d'identification SMTP qui sont dérivées d'informations d'identification de sécurité temporaires.

Amazon Simple Storage Service

Amazon S3 prend en charge l'autorisation basée sur des balises pour les compartiments de répertoires, les ressources d'objets, Amazon S3 Storage Lens, les groupes Storage Lens et les points d'accès.

Amazon S3 prend en charge les rôles liés au service pour Amazon S3 Storage Lens.

AWS Trusted Advisor

L'accès à l'API Trusted Advisor se fait via l' Support API et est contrôlé par les politiques Support IAM.

Amazon Virtual Private Cloud

Amazon VPC prend en charge l'attachement d'une seule politique de ressource à un point de terminaison d'un VPC afin de limiter les éléments accessibles via ce point de terminaison. Pour plus

d'informations sur l'utilisation des politiques basées sur les ressources pour contrôler l'accès à celles-ci à partir de points de terminaison Amazon VPC spécifiques, consultez Control d'accès à l'utilisation des politiques de point de terminaison dans le Guide AWS PrivateLink .

AWS X-Ray

X-Ray ne prend pas en charge les autorisations au niveau des ressources pour toutes les actions.

X-Ray prend en charge le contrôle d'accès basé sur des balises pour les groupes et les règles d'échantillonnage.

AWS Signature Version 4 pour les demandes d'API

> **Important**
>
> Si vous utilisez un outil AWS SDK (voir Exemples de code et bibliothèques) ou AWS Command Line Interface (AWS CLI) pour envoyer des demandes d'API AWS, vous pouvez ignorer le processus de signature, car les clients du SDK et de la CLI authentifient vos demandes à l'aide des clés d'accès que vous fournissez. À moins que vous n'ayez une bonne raison de ne pas le faire, nous vous recommandons de toujours utiliser un kit SDK ou la CLI. Dans les régions qui prennent en charge plusieurs versions de signature, les demandes de signature manuelle impliquent que vous devez spécifier la version de signature à utiliser. Lorsque vous envoyez des demandes à des points Multi-Region d'accès, les SDK et l'interface de ligne de commande passent automatiquement à la version 4A de Signature sans configuration supplémentaire.

Les informations d'authentification que vous envoyez dans une demande doivent inclure une signature. AWS Signature Version 4 (Sigv4) est le protocole de AWS signature permettant d'ajouter des informations d'authentification aux demandes d' AWS API.

Vous n'utilisez pas votre clé d'accès secrète pour signer les requêtes d'API. Vous utilisez plutôt le processus de signature SigV4. La signature des requêtes implique :

1. Création d'une requête canonique basée sur les détails de la requête.
2. Calcul d'une signature à l'aide de vos AWS informations d'identification.
3. Ajout de cette signature à la requête en tant qu'en-tête Autorisation.

AWS reproduit ensuite ce processus et vérifie la signature, en accordant ou en refusant l'accès en conséquence.

Symmetric SIGv4 nécessite que vous dériviez une clé qui est limitée à un seul AWS service, dans une seule AWS région, un jour donné. La clé et la signature calculée sont donc différentes pour chaque région, ce qui signifie que vous devez connaître la région à laquelle la signature est destinée.

Asymmetric Signature Version 4 (SigV4a) est une extension qui permet de signer avec un nouvel algorithme et de générer des signatures individuelles vérifiables dans plusieurs Régions AWS . Avec SigV4A, vous pouvez signer une demande pour plusieurs régions, avec un routage et un basculement fluides entre les régions. Lorsque vous utilisez le AWS SDK ou AWS CLI que vous invoquez une fonctionnalité qui nécessite une signature multirégionale, le type de signature est automatiquement modifié pour utiliser SigV4a. Pour en savoir plus, consultez Comment ? AWS SigV4a fonctionne.

Comment ? AWS SiGv4 fonctionne

Les étapes suivantes décrivent le processus général de calcul d'une signature à l'aide de SigV4 :

1. La chaîne à signer dépend du type de demande. Par exemple, lorsque vous utilisez l'en-tête d'autorisation HTTP ou les paramètres de requête pour l'authentification, vous utilisez une combinaison d'éléments de requête pour créer la chaîne à signer. Pour une requête HTTP POST, la politique `POST` de la demande est la chaîne que vous signez.
2. La clé de signature est une série de calculs, le résultat de chaque étape étant intégré dans la suivante. La dernière étape est la clé de signature.
3. Lorsqu'un AWS service reçoit une demande authentifiée, il recrée la signature à l'aide des informations d'authentification contenues dans la demande. Si les signatures correspondent, le service traite la demande. Sinon, il rejette la demande.

Pour de plus amples informations, veuillez consulter Les éléments d'un AWS Signature de la demande d'API.

Comment ? AWS SigV4a fonctionne

Sigv4a utilise des signatures asymétriques basées sur le chiffrement à clé publique-privée. SigV4a passe par un processus de dérivation d'informations d'identification d'une portée similaire à celui de SIGv4, sauf que SigV4a utilise la même clé pour signer toutes les demandes sans avoir à dériver une

clé de signature distincte en fonction de la date, du service et de la région. Une paire de clés ECDSA (Elliptic Curve Digital Signature Algorithm) peut être dérivée de votre clé d'accès secrète existante. AWS

Le système utilise le chiffrement asymétrique pour vérifier les signatures multirégions, de sorte qu' AWS n'a besoin de stocker que vos clés publiques. Les clés publiques ne sont pas secrètes et ne peuvent pas être utilisées pour signer des requêtes. Des signatures asymétriques sont requises pour les demandes d'API multirégionales, par exemple avec les points d' Multi-Region accès Amazon S3.

Les étapes suivantes décrivent le processus général de calcul d'une signature à l'aide de SigV4a :

1. La chaîne à signer dépend du type de demande. Par exemple, lorsque vous utilisez l'en-tête d'autorisation HTTP ou les paramètres de requête pour l'authentification, vous utilisez une combinaison d'éléments de requête pour créer la chaîne à signer. Pour une requête HTTP POST, la politique `POST` de la demande est la chaîne que vous signez.
2. La clé de signature est dérivée d'une clé d'accès secrète AWS par le biais d'une série de calculs, le résultat de chaque étape étant intégré dans la suivante. La dernière étape produit la paire de clés.
3. Lorsqu'un AWS service reçoit une demande signée avec SigV4a, il AWS vérifie la signature en utilisant uniquement la moitié publique de la paire de clés. Si la signature est valide, la requête est authentifiée et le service la traite. Les demandes dont les signatures ne sont pas valides sont rejetées.

Pour plus d'informations sur SigV4a pour les requêtes d'API multi-régions, consultez le projet https://github.com/aws-samples/sigv4a-signing-examples sigv4a-signing-examples sur. GitHub

Quand signer des demandes ?

Lorsque vous écrivez un code personnalisé qui envoie des demandes d'API à AWS, vous devez inclure un code qui signe les demandes. Vous pouvez écrire du code personnalisé pour les raisons suivantes :

- Vous utilisez un langage de programmation pour lequel il n'existe aucun kit SDK AWS .
- Vous devez avoir un contrôle total sur la façon dont les demandes sont envoyées à AWS.

Alors que les demandes d'API authentifient l'accès avec AWS Sigv4, AWS les SDK AWS CLI authentifient également vos demandes à l'aide des clés d'accès que vous fournissez. Pour plus

d'informations sur l'authentification à l'aide des AWS kits SDK et du AWS CLI, consultez. Ressources supplémentaires

Pourquoi les demandes sont-elles signées ?

Le processus de signature aide à sécuriser les demandes de différentes façons :

- Vérifier l'identité du demandeur

  Les demandes authentifiées nécessitent une signature que vous créez à l'aide de vos clés d'accès (ID de clé d'accès, clé d'accès secrète). Si vous utilisez des informations d'identification de sécurité temporaires, les calculs de signature nécessitent également un jeton de sécurité. Pour plus d'informations, veuillez consulter la rubrique Accès par programmation des informations d'identification de sécuritéAWS.

- Protéger les données en transit

  Pour éviter qu'une demande ne soit falsifiée pendant son transit, certains de ses éléments sont utilisés pour calculer son hachage (digest) et la valeur de hachage obtenue est incluse comme partie intégrante de la demande. Lorsqu'un Service AWS reçoit la demande, il utilise les mêmes informations pour calculer un hachage et le compare à la valeur de hachage de votre demande. Si les valeurs ne correspondent pas, AWS refuse la demande.

- Assurer une protection contre les attaques potentielles par relecture

  Dans la plupart des cas, une demande doit parvenir AWS dans les cinq minutes suivant l'horodatage indiqué dans la demande. Dans le cas contraire, AWS refuse la demande.

AWS SigV4 peut être exprimé dans l'en-tête HTTP Authorization ou sous forme de chaîne de requête dans l'URL. Pour de plus amples informations, veuillez consulter Méthodes d'authentification.

Ressources supplémentaires

- Pour plus d'informations sur le processus de signature SigV4 pour les différents services, consultez Demander des exemples de signature.
- Pour configurer les informations d'identification pour l'accès programmatique à la AWS CLI, voir Authentification et informations d'accès dans le Guide de l'utilisateur de l'interface de ligne de AWS commande.

- Les AWS kits SDK incluent le code source activé GitHub pour la signature des demandes AWS d'API. Pour obtenir des exemples de code, veuillez consulter Exemples de projets dans AWS référentiel d'échantillons.
  - AWS SDK pour .NET – AWSSigV4Signer.cs
  - AWS SDK pour C++ – AWSAuthV4Signer.cpp
  - AWS SDK pour Go — sigv4.go
  - AWS SDK pour Java – BaseAws4Signer.java
  - AWS SDK pour JavaScript — signature-v4
  - AWS SDK pour PHP – SignatureV4.php
  - AWS SDK pour Python (Boto) — signers.py
  - AWS SDK pour Ruby — signer.rb
- Si vous rencontrez des erreurs lors de la signature des demandes, consultezRésoudre les problèmes liés à la signature de Signature Version 4 pour AWS Demandes d'API.

Les éléments d'un AWS Signature de la demande d'API

> **Important**
>
> À moins que vous n'utilisiez les AWS SDK ou l'interface de ligne de commande, vous devez écrire du code pour calculer les signatures qui fournissent des informations d'authentification dans vos demandes. Le calcul des AWS signatures dans Signature Version 4 peut s'avérer complexe, et nous vous recommandons d'utiliser les AWS SDK ou l'interface de ligne de commande dans la mesure du possible.

Chaque HTTP/HTTPS demande utilisant la signature Signature Version 4 doit contenir ces éléments.

Eléments

- Spécification du point de terminaison
- Action
- Paramètres d'action
- Date
- Informations d'authentification

Spécification du point de terminaison

Spécifie le nom DNS du point de terminaison auquel vous envoyez la requête. Ce nom contient généralement le code du service et la région. Par exemple, le nom du point de terminaison Amazon DynamoDB de la région `us-east-1` est `dynamodb.us-east-1.amazonaws.com`.

Pour les HTTP/1.1 demandes, vous devez inclure l'`Host`en-tête. Pour les HTTP/2 demandes, vous pouvez inclure l'`:authority`en-tête ou l'`Host`en-tête. Utilisez uniquement l'`:authority`en-tête pour vous conformer à la HTTP/2 spécification. Les HTTP/2 demandes ne sont pas prises en charge par tous les services.

Pour les points de terminaison pris en charge par chaque service, consultez Points de terminaison et quotas de service dans Références générales AWS.

Action

Spécifie une action d'API pour le service. Par exemple, l'action DynamoDB `CreateTable` ou l'action Amazon EC2 `DescribeInstances`.

Pour les actions prises en charge par chaque service, consultez la référence d'autorisation des services.

Paramètres d'action

Spécifie les paramètres de l'action spécifiée dans la requête. Chaque action d' AWS API possède un ensemble de paramètres obligatoires et facultatifs. La version de l'API est généralement un paramètre obligatoire.

Pour connaître les paramètres pris en charge par une action d'API, consultez la référence d'API du service.

Date

Spécifie la date et l'heure de la requête. L'inclusion de la date et de l'heure dans la requête permet d'empêcher des tiers d'intercepter cette dernière et de la soumettre à nouveau ultérieurement. La date que vous spécifiez dans l'étendue des informations d'identification doit correspondre à la date de votre requête.

L'horodatage doit être au format UTC et au format ISO 8601 suivant : AAAAMMJJTHHMMSSZ. Par exemple, `20220830T123600Z`. N'incluez pas de millisecondes dans l'horodatage.

Vous pouvez utiliser un en-tête `date` ou `x-amz-date`, ou inclure `x-amz-date` comme un paramètre de requête. Si nous ne trouvons pas d'en-tête `x-amz-date`, nous cherchons un en-tête `date`.

Informations d'authentification

Chaque demande que vous envoyez doit inclure les informations suivantes. AWS utilise ces informations pour garantir la validité et l'authenticité de la demande.

- Algorithme : algorithme que vous utilisez dans le cadre du processus de signature.
  - SigV4 : `AWS4-HMAC-SHA256` permet de spécifier la version 4 de la Signature à l'aide de l'algorithme de hachage `HMAC-SHA256`.
  - SigV4a : utilisez `AWS4-ECDSA-P256-SHA256` pour spécifier l'algorithme de hachage `ECDSA-P256-SHA-256`.
- Informations d'identification : chaîne formée en concaténant votre ID de clé d'accès et les composants de votre champ d'application d'informations d'identification.
  - SigV4 : la portée des informations d'identification comprend votre clé d'accès, la date au format AAAAMMJJ, le code de région, le code de service et la chaîne de terminaison `aws4_request`, séparés par des barres obliques (/). Le code Région, le code de service et la chaîne de terminaison doivent utiliser des caractères minuscules.

    ```
    AKIAIOSFODNN7EXAMPLE/YYYYMMDD/region/service/aws4_request
    ```

  - SigV4a : la portée des informations d'identification comprend la date au format AAAAMMJJ, le nom du service et la chaîne de terminaison `aws4_request`, séparés par des barres obliques (/). Notez que la portée des informations d'identification n'inclut pas la région, car celle-ci est couverte dans un en-tête `X-Amz-Region-Set` distinct.

    ```
    AKIAIOSFODNN7EXAMPLE/YYYYMMDD/service/aws4_request
    ```

- En-têtes signés : en-têtes HTTP à inclure dans la signature, séparés par des points-virgules (;). Par exemple, `host;x-amz-date`.

  Pour SigV4a, vous devez inclure un en-tête de jeu de régions qui spécifie l'ensemble des régions dans lesquelles la demande sera valide. L'en-tête `X-Amz-Region-Set` est spécifié sous forme de liste de valeurs séparées par des virgules. L'exemple suivant montre un en-tête de région qui permet d'effectuer une demande dans les régions us-east-1 et us-west-1.

```
X-Amz-Region-Set=us-east-1,us-west-1
```

Vous pouvez utiliser des caractères génériques (*) dans les régions pour spécifier plusieurs régions. Dans l'exemple suivant, l'en-tête permet de faire une demande à la fois dans les régions us-west-1 et us-west-2.

```
X-Amz-Region-Set=us-west-*
```

- Signature : chaîne codée en hexadécimal qui représente la signature calculée. Vous devez calculer la signature en utilisant l'algorithme que vous avez spécifié dans le paramètre `Algorithm`.

Pour de plus amples informations, veuillez consulter Méthodes d'authentification.

Méthodes d'authentification

> **Important**
>
> À moins que vous n'utilisiez les AWS SDK ou l'interface de ligne de commande, vous devez écrire du code pour calculer les signatures qui fournissent des informations d'authentification dans vos demandes. Le calcul des AWS signatures dans Signature Version 4 peut s'avérer complexe, et nous vous recommandons d'utiliser les AWS SDK ou l'interface de ligne de commande dans la mesure du possible.

Vous pouvez exprimer les informations d'authentification à l'aide de l'une des méthodes suivantes.

En-tête d'autorisation HTTP

L'en-tête HTTP `Authorization` est l'une des méthodes d'authentification des demandes les plus courantes. Toutes les opérations de l'API REST (à l'exception des chargements via un navigateur utilisant des demandes `POST`) nécessitent cet en-tête.

Les exemples suivants montrent la valeur d'en-tête `Authorization` pour SigV4 et SigV4a. Des sauts de ligne sont ajoutés à cet exemple pour faciliter la lecture. Dans votre code, l'en-tête doit être une chaîne continue. Il n'y a pas de virgule entre l'algorithme et les informations d'identification, mais les autres éléments doivent être séparés par des virgules.

Example SigV4

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024
```

Example SigV4a

```
Authorization: AWS4-ECDSA-P256-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/s3/aws4_request,
SignedHeaders=host;range;x-amz-date;x-amz-region-set,
Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024
```

Le tableau suivant décrit les différents composants de la valeur de l'en-tête d'autorisation dans l'exemple précédent :

Composant	Description
Autorisation	Algorithme utilisé pour calculer la signature. • SigV4 – Utilisez `AWS4-HMAC-SHA256` . Cette chaîne identifie AWS SIGv4 (`AWS4`) et l'`HMAC-SHA256` algorithme. • SigV4A – Utilisez `AWS4-ECDSA-P256-SHA256` . Cette chaîne identifie AWS SIGv4 (`AWS4`) et l'`ECDSA-P256-SHA-256` algorithme.
Informations d'identification	Votre ID de clé d'accès et les informations relatives au champ d'application. • SigV4 : inclut la date, la région et le service utilisés pour calculer la Signature. Cette chaîne présente le format suivant : `<your-access-key-id>/<date>/ <aws-region>/<aws-service>/ aws4_request`

Composant	Description
	• SigV4a : inclut la date et le service utilisés pour calculer la Signature. Cette chaîne présente le format suivant : `<your-access-key-id>/<date>/ <aws-service>/aws4_request` La valeur **<date>** est spécifiée à l'aide du format AAAAMMJJ. La valeur **<aws-service>** est S3 lors de l'envoi d'une demande à Amazon S3.
SignedHeaders	Une liste d'en-têtes de demande séparés par des points-virgules que vous avez utilisés pour calculer Signature. La liste inclut uniquemen t les noms d'en-tête, qui doivent être en minuscules. Par exemple : `host;range;x-amz-date` Pour SigV4a, vous devez inclure un en-tête de jeu de régions qui spécifie l'ensemble des régions dans lesquelles la demande sera valide. L'en-tête X-Amz-Region-Set est spécifié sous forme de liste de valeurs séparées par des virgules.
Signature	La signature de 256 bits exprimée en 64 caractères hexadécimaux minuscule s. Exemples :`fe5f80f77d5fa3beca 038a248ff027d0445342fe2855d dc963176630326f1024` Veuillez noter que les calculs de signature varient en fonction de l'option que vous choisissez pour transférer la charge utile.

Paramètres des chaînes de requête

Vous pouvez utiliser une chaîne de requête pour exprimer entièrement une demande dans une URL. Dans ce cas, vous utilisez les paramètres de requête pour fournir les informations de demande, y compris les informations d'authentification. Comme la signature de la demande fait partie de l'URL, ce type d'URL est généralement appelé URL présignée. Vous pouvez utiliser des URL présignées pour intégrer des liens cliquables au format HTML, qui peuvent être valides jusqu'à sept jours. Pour plus d'informations, consultez Authentification des demandes : utilisation des paramètres de requête (AWS Signature version 4) dans le manuel Amazon S3 API Reference.

Les exemples suivants montrent des URL pré-Signées pour SigV4 et SigV4a. Des sauts de ligne sont ajoutés à cet exemple pour faciliter la lecture :

Example SigV4

```
https://s3.amazonaws.com/amzn-s3-demo-bucket/test.txt ?
X-Amz-Algorithm=AWS4-HMAC-SHA256 &
X-Amz-Credential=<your-access-key-id>/20130721/<region>/s3/aws4_request &
X-Amz-Date=20130721T201207Z &
X-Amz-Expires=86400 &
X-Amz-SignedHeaders=host &X-Amz-Signature=<signature-value>
```

Example SigV4a

```
https://s3.amazonaws.com/amzn-s3-demo-bucket/test.txt ?
X-Amz-Algorithm=AWS4-ECDSA-P256-SHA256 &
X-Amz-Credential=<your-access-key-id>/20240721/s3/aws4_request &
X-amz-Region-Set=<regionset> &
X-Amz-Date=20240721T201207Z &
X-Amz-Expires=86400 &
X-Amz-SignedHeaders=host;x-amz-region-set &
X-Amz-Signature=<signature-value>
```

> **Note**
>
> La valeur `X-Amz-Credential` dans l'URL affiche le caractère « / » uniquement pour des raisons de lisibilité. En pratique, il doit être encodé sous la forme %2F. Par exemple : `&X-Amz-Credential=<your-access-key-id>%2F20130721%2Fus-east-1%2Fs3%2Faws4_request`

Le tableau suivant décrit les paramètres de requête de l'URL qui fournissent des informations d'authentification.

Nom du paramètre de la chaîne d'interrogation	Description
X-Amz-Algorithm	Version de la AWS signature et algorithme que vous avez utilisés pour calculer la signature. • SigV4 – Utilisez `AWS4-HMAC-SHA256` . Cette chaîne identifie AWS SIGv4 (`AWS4`) et l'`HMAC-SHA256` algorithme. • SigV4A – Utilisez `AWS4-ECDSA-P256-SHA256` . Cette chaîne identifie AWS SIGv4 (`AWS4`) et l'`ECDSA-P256-SHA-256` algorithme.
X-Amz-Credential	Outre l’ID de votre clé d’accès, ce paramètre fournit également le champ d’application pour lequel la signature est valide. Cette valeur doit correspondre au champ d'application que vous utilisez dans les calculs de signature, comme indiqué dans la section suivante. • SigV4 : le format général de cette valeur de paramètre est le suivant : `<your-access-key-id>/<date>/<AWS Region>/<AWS-service>/aws4_request` Par exemple : `AKIAIOSFODNN7EXAMPLE/20130721/us-east-1/s3/aws4_request` • SigV4a : le format général de cette valeur de paramètre est le suivant : `<your-access-key-id>/<date>/<AWS-service>/aws4_request` Par exemple : `AKIAIOSFODNN7EXAMPLE/20130721/s3/aws4_request`

Nom du paramètre de la chaîne d'interrogation	Description
	La région pour SigV4a est définie dans l'en-tête de l'ensemble de régions `X-Amz-Region-Set` . Pour une liste des chaînes AWS régionales, voir Points de terminaison régionaux dans la référence AWS générale.
X-Amz-Region-Set	Ensemble de régions dans lesquelles la demande sera valide. L'en-tête x-amz-region-set est spécifié sous forme de liste de valeurs séparées par des virgules.
X-Amz-Date	Le format de date et d'heure doit respecter la norme ISO 8601 et doit être au format `yyyyMMddTHHmmssZ` . Par exemple, si la date et l'heure étaient « 08/01 /2016 15:32:41. 982-700 », elles doivent d'abord être convertie s en UTC (temps universel coordonné), puis soumises sous la forme « 20160801T 223241Z ».
X-Amz-Expires	Indique la période, en secondes, pendant laquelle l'URL présignée générée est valide. Par exemple, 86 400 (24 heures). Cette valeur est un nombre entier. La valeur minimale que vous pouvez définir est 1 et la valeur maximale est de 604 800 (sept jours). Une URL présignée peut être valide pendant sept jours au maximum, car la clé de signature que vous utilisez pour le calcul de la signature est valide pendant sept jours au maximum.

Nom du paramètre de la chaîne d'interrogation	Description
X-Amz-SignedHeaders	Répertorie les en-têtes que vous avez utilisés pour calculer la signature. Les en-têtes suivants sont obligatoires pour les calculs de signature : • L'en-tête de l'hôte HTTP. • Tous les en-têtes x-amz-* que vous prévoyez d'ajouter à la demande. • Pour SigV4a, X-Amz-Region-Set il est nécessaire de spécifier les régions dans lesquelles la demande peut être faite. Pour plus de sécurité, vous devez signer tous les en-têtes de demande que vous prévoyez d'inclure dans votre demande.
X-Amz-Signature	Fournit la signature pour authentifier votre demande. Cette signature doit correspondre à la signature que le service calcule, sinon, le service refuse la demande. Par exemple, `733255ef022bec3f2a8701cd61d4b371f3f28c9f193a1f02279211d48d5193d7` Les calculs de signature sont décrits dans la section suivante.
X-Amz-Security-Token	Paramètre d'information d'identification facultatif si vous utilisez des informations d'identification provenant du service STS.

Créer un document signé AWS Requête d'API

> **Important**
>
> Si vous utilisez un outil AWS SDK (voir Exemples de code et bibliothèques) ou AWS Command Line Interface (AWS CLI) pour envoyer des requêtes d'API AWS, vous pouvez ignorer cette section car les clients du SDK et de la CLI authentifient vos demandes à l'aide des clés d'accès que vous fournissez. À moins que vous n'ayez une bonne raison de ne pas le faire, nous vous recommandons de toujours utiliser un kit SDK ou la CLI.
> Dans les régions qui prennent en charge plusieurs versions de signature, les demandes de signature manuelle impliquent que vous devez spécifier la version de signature utilisée. Lorsque vous envoyez des demandes à des points Multi-Region d'accès, les SDK et l'interface de ligne de commande passent automatiquement à la version 4A de Signature sans configuration supplémentaire.

Vous pouvez utiliser le protocole de signature AWS SIGv4 pour créer une demande signée pour les demandes AWS d'API.

1. Création d'une requête canonique basée sur les détails de la requête.
2. Calcul d'une signature à l'aide de vos AWS informations d'identification.
3. Ajout de cette signature à la requête en tant qu'en-tête Autorisation.

AWS reproduit ensuite ce processus et vérifie la signature, en accordant ou en refusant l'accès en conséquence.

Pour savoir comment utiliser AWS Sigv4 pour signer des demandes d'API, consultezDemander des exemples de signature.

Le tableau suivant décrit les fonctions utilisées dans le processus de création d'une demande signée. Vous devez implémenter du code pour ces fonctions. Pour plus d'informations, consultez les exemples de code dans les SDK AWS.

Fonction	Description
`Lowercase()`	Convertit la chaîne en minuscules.

Fonction	Description
`Hex()`	Encodage en minuscules en base 16.
`SHA256Hash()`	Fonction de hachage cryptographique Secure Hash Algorithm (SHA, algorithme de hachage sécurisé).
`HMAC-SHA256()`	Calcule HMAC à l'aide de l'algorithme SHA256 avec la clé de signature fournie. Il s'agit de la Signature finale lorsque vous Signez avec Sigv4.
`ECDSA-Sign`	Signature ECDSA (Elliptic Curve Digital Signature Algorithm) calculée à l'aide de signatures asymétriques basées sur le chiffrement à clé publique-privée.
`KDF(K, Label, Context, L)`	Un SP800-108 KDF NIST en mode compteur utilisant la fonction PRF telle HMAC-SHA256 que définie dans le NIST SP 800-108r1.
`Oct2Int(byte[ ])`	Une fonction octet à entier telle que décrite dans la norme X9.62 ANSI.
`Trim()`	Supprimez tous les espaces blancs de début ou de fin.

Fonction	Description
`UriEncode()`	L'URI code chaque octet. UriEncode() doit appliquer les règles suivantes : • L'URI encode tous les octets sauf les caractères non réservés : A-Z, a-z, 0-9, -, ., _ et ~. • Le caractère d'espace est un caractère réservé qui doit être codé sous la forme « %20 » (et non sous la forme « + »). • Chaque octet encodé par URI est formé par un « % » et la valeur hexadécimale à deux chiffres de l'octet. • Les lettres de la valeur hexadécimale doivent être en majuscules, par exemple « %1A ». • Encodez la barre oblique « / » partout sauf dans le nom de la clé de l'objet. Par exemple, si le nom de clé de l'objet est `photos/Jan/sample.jpg` , la barre oblique du nom de la clé n'est pas encodée. **Important** Les UriEncode fonctions standard fournies par votre plate-forme de développement peuvent ne pas fonctionner en raison de différences d'implémentation et de l'ambiguïté associée dans les RFC sous-jacents. Nous vous recommandons d'écrire votre propre UriEncode fonction personnalisée pour vous assurer que votre encodage fonctionnera.

Fonction	Description
	Pour voir un exemple de UriEncode fonction en Java, voir Java Utilities sur le GitHub site Web.

> **Note**
>
> Lorsque vous signez vos demandes, vous pouvez utiliser AWS SIGv4 ou AWS SigV4a. La principale différence entre les deux est déterminée par la façon dont la signature est calculée. Avec SigV4a, l'ensemble de régions est inclus dans la chaîne à Signer, mais ne fait pas partie de l'étape de dérivation des informations d'identification.

Requêtes de signature avec des informations d'identification de sécurité temporaires

Au lieu d'utiliser des informations d'identification à long terme pour signer une demande, vous pouvez utiliser des informations d'identification de sécurité temporaires fournies par AWS Security Token Service (AWS STS).

Lorsque vous utilisez des informations d'identification de sécurité temporaires, vous devez ajouter `X-Amz-Security-Token` à l'en-tête Autorisation ou l'inclure à la chaîne de requête pour contenir le jeton de session. Certains services nécessitent que vous ajoutiez `X-Amz-Security-Token` à la requête canonique. Pour les autres services, vous n'avez qu'à ajouter `X-Amz-Security-Token` à la fin, une fois que vous avez calculé la signature. Consultez la documentation de chacun Service AWS pour connaître les exigences spécifiques.

Récapitulatif des étapes de signature

Création d'une requête canonique

Organisez le contenu de votre demande (hôte, action, en-têtes, etc.) dans un format standard canonique. La requête canonique est l'une des entrées utilisées pour créer la chaîne de connexion. Pour plus de détails sur la création de la requête canonique, consultez Les éléments d'un AWS Signature de la demande d'API.

Création d'un hachage de la requête canonique

Hashez la requête canonique avec le même algorithme que celui que vous avez utilisé pour créer le hachage des données utiles. Le hachage de la requête canonique est une chaîne de caractères hexadécimaux en minuscules.

Création d'une chaîne à signer

Créez une chaîne de connexion avec la requête canonique et des informations supplémentaires telles que l'algorithme, la date de la requête, la portée des informations d'identification et le hachage de la requête canonique.

Dérivation d'une clé de signature

Utilisez la clé d'accès secrète pour dériver la clé utilisée pour signer la demande.

Calcul de la signature

Effectuez une opération de hachage à clé sur la chaîne à signer en utilisant la clé de signature dérivée comme clé de hachage.

Ajout d'une signature à la requête

Ajouter la signature calculée à un en-tête HTTP ou à la chaîne de requête de la requête.

Création d'une requête canonique

Pour créer une requête canonique, concaténez les chaînes suivantes, séparées par des caractères de saut de ligne. Cela permet de s'assurer que la signature que vous calculez peut correspondre à la signature AWS calculée.

```
<HTTPMethod>\n
<CanonicalURI>\n
<CanonicalQueryString>\n
<CanonicalHeaders>\n
<SignedHeaders>\n
<HashedPayload>
```

- *HTTPMethod*— La méthode HTTP, telle que `GET`PUT,`HEAD`, et`DELETE`.
- *CanonicalUri*— URI-encoded Version de l'URI du composant de chemin absolu, en commençant par `/` celle qui suit le nom de domaine et jusqu'à la fin de la chaîne ou jusqu'au point d'interrogation (`?`) si vous avez des paramètres de chaîne de requête. Si le chemin d'accès absolu

est vide, utilisez un caractère barre oblique (/). L'URI de l'exemple suivant, `/amzn-s3-demo-bucket/myphoto.jpg`, est le chemin absolu et vous n'encodez pas la / dans le chemin absolu :

```
http://s3.amazonaws.com/amzn-s3-demo-bucket/myphoto.jpg
```

- *CanonicalQueryString*— Les paramètres de la chaîne de URI-encoded requête. Vous URI-encode nommez et valorisez individuellement. Vous devez également trier les paramètres de la chaîne de requête canonique par ordre alphabétique par nom de clé. Le tri s'effectue après l'encodage. La chaîne de requête dans l'exemple d'URI suivant est la suivante :

  ```
  http://s3.amazonaws.com/amzn-s3-demo-bucket?prefix=somePrefix&marker=someMarker&max-keys=2
  ```

  La chaîne de requête canonique est la suivante (des sauts de ligne sont ajoutés à cet exemple pour faciliter la lecture) :

  ```
  UriEncode("marker")+"="+UriEncode("someMarker")+"&"+
  UriEncode("max-keys")+"="+UriEncode("20") + "&" +
  UriEncode("prefix")+"="+UriEncode("somePrefix")
  ```

  Lorsqu'une requête cible une sous-ressource, la valeur du paramètre de requête correspondant est une chaîne vide (""). Par exemple, l'URI suivant identifie la sous-ressource `ACL` sur le compartiment `amzn-s3-demo-bucket` :

  ```
  http://s3.amazonaws.com/amzn-s3-demo-bucket?acl
  ```

  Dans ce cas, il s' CanonicalQueryString agirait de :

  ```
  UriEncode("acl") + "=" + ""
  ```

  Si l'URI ne contient pas de ?, la requête ne contient aucune chaîne de requête et vous définissez la chaîne de requête canonique sur une chaîne vide (""). Vous devrez tout de même inclure le caractère de saut de ligne ("\n").

- *CanonicalHeaders*— Liste des en-têtes de demande avec leurs valeurs. Les paires de nom et de valeur d'en-tête individuelles sont séparées par le caractère de saut de ligne ("\n"). Voici un exemple de CanonicalHeader :

```
Lowercase(<HeaderName1>)+":"+Trim(<value>)+"\n"
Lowercase(<HeaderName2>)+":"+Trim(<value>)+"\n"
...
Lowercase(<HeaderNameN>)+":"+Trim(<value>)+"\n"
```

CanonicalHeaders la liste doit inclure les éléments suivants :

- En-tête `host` HTTP.
- Si l'`Content-Type`en-tête est présent dans la demande, vous devez l'ajouter à la *`CanonicalHeaders`* liste.
- Les en-têtes `x-amz-*` que vous prévoyez d'inclure dans votre demande doivent également être ajoutés. Par exemple, si vous utilisez des informations d'identification de sécurité temporaires, vous devez inclure `x-amz-security-token` dans votre demande. Vous devez ajouter cet en-tête dans la liste des*`CanonicalHeaders`*.
- Pour SigV4a, vous devez inclure un en-tête de jeu de régions qui spécifie l'ensemble des régions dans lesquelles la demande sera valide. L'en-tête `X-Amz-Region-Set` est spécifié sous forme de liste de valeurs séparées par des virgules. L'exemple suivant montre un en-tête de région qui permet d'effectuer une demande dans les régions us-east-1 et us-west-1.

  ```
  X-Amz-Region-Set=us-east-1,us-west-1
  ```

  Vous pouvez utiliser des caractères génériques (*) dans les régions pour spécifier plusieurs régions. Dans l'exemple suivant, l'en-tête permet de faire une demande à la fois dans les régions us-west-1 et us-west-2.

  ```
  X-Amz-Region-Set=us-west-*
  ```

> **Note**
>
> L'en-tête `x-amz-content-sha256` est obligatoire pour les requêtes AWS Amazon S3. Il fournit un hachage de la charge utile de la requête. S'il n'y a pas de charge utile, vous devez fournir le hachage d'une chaîne vide.

Chaque nom d'en-tête doit :

- utiliser des caractères minuscules ;
- apparaître par ordre alphabétique ;

- être suivi de deux points (`:`).

Pour les valeurs, vous devez :

- supprimer toutes les espaces de début ou de fin ;
- convertir des espaces séquentielles en une espace unique ;
- séparez les valeurs d'un en-tête à valeurs multiples par des virgules.
- Vous devez inclure l'en-tête host (HTTP/1.1) ou l'en-tête:authority (HTTP/2), ainsi que tous `x-amz-*` les en-têtes de la signature. Vous pouvez éventuellement inclure d'autres en-têtes standard dans la signature, tels que content-type.

Les fonctions `Lowercase()` et `Trim()` utilisées dans cet exemple sont décrites dans la section précédente.

L'exemple suivant est une chaîne `CanonicalHeaders`. Les noms d'en-tête sont en minuscules et sont triés.

```
host:s3.amazonaws.com
x-amz-content-sha256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
x-amz-date:20130708T220855Z
```

> **Note**
>
> Aux fins du calcul d'une signature d'autorisation, seuls l'hôte et les en-têtes `x-amz-*` sont obligatoires. Toutefois, afin d'éviter toute altération des données, vous devez envisager d'inclure les en-têtes supplémentaires dans le calcul de la signature.
> N'incluez pas les en-têtes tronçon qui sont fréquemment modifiés lors du transit à travers un système complexe. Cela inclut tous les en-têtes de transport volatils qui sont modifiés par les proxys, les équilibreurs de charge et les nœuds d'un système distribué, y compris `connection`, `x-amzn-trace-id`, `user-agent`, `keep-alive`, `transfer-encoding`, `TE`, `trailer`, `upgrade`, `proxy-authorization`, et `proxy-authenticate`.

- *`SignedHeaders`*— Liste de noms d'en-têtes de demande en minuscules, triée par ordre alphabétique et séparés par des points-virgules. Les en-têtes de demande de la liste sont les mêmes que ceux que vous avez inclus dans la chaîne `CanonicalHeaders`. Pour l'exemple précédent, la valeur de *`SignedHeaders`* serait la suivante :

```
host;x-amz-content-sha256;x-amz-date
```

- *HashedPayload*— Chaîne créée en utilisant la charge utile contenue dans le corps de la requête HTTP comme entrée d'une fonction de hachage. Cette chaîne utilise des caractères hexadécimaux minuscules.

  ```
  Hex(SHA256Hash(<payload>))
  ```

  Si la requête ne contient pas de données utiles, vous calculez un hachage de la chaîne vide, comme lorsque vous récupérez un objet à l'aide d'une requête GET, il n'y a rien dans les données utiles.

  ```
  Hex(SHA256Hash(""))
  ```

  > **Note**
  >
  > Pour Amazon S3, incluez la chaîne littérale UNSIGNED-PAYLOAD lors de la création d'une requête canonique, et définissez la même valeur que celle de l'en-tête x-amz-content-sha256 lors de l'envoi de la requête.
  > Hex(SHA256Hash("UNSIGNED-PAYLOAD"))

Création d'un hachage de la requête canonique

Créez un hachage (digest) de la requête canonique avec le même algorithme que celui que vous avez utilisé pour créer le hachage de la charge utile. Le hachage de la requête canonique est une chaîne de caractères hexadécimaux en minuscules.

Création d'une chaîne à signer

Pour créer une chaîne, concaténez les chaînes suivantes, séparées par des sauts de ligne. Ne terminez pas cette chaîne par un caractère de saut de ligne.

```
Algorithm \n
RequestDateTime \n
CredentialScope  \n
HashedCanonicalRequest
```

- *Algorithm*— Algorithme utilisé pour créer le hachage de la requête canonique.
  - SigV4 : utilisez `AWS4-HMAC-SHA256` pour spécifier l'algorithme de hachage `HMAC-SHA256`.
  - SigV4a : utilisez `AWS4-ECDSA-P256-SHA256` pour spécifier l'algorithme de hachage `ECDSA-P256-SHA-256`.
- *RequestDateTime*— La date et l'heure utilisées dans la portée des informations d'identification. Cette valeur est l'heure UTC actuelle au format ISO 8601 (par exemple, `20130524T000000Z`).
- *CredentialScope*— La portée des informations d'identification, qui limite la signature résultante à la région et au service spécifiés.
  - SigV4 : les informations d'identification comprennent votre clé d'accès, la date au format `YYYYMMDD`, le code de région, le code de service et la chaîne de terminaison `aws4_request`, séparés par des barres obliques (/). Le code Région, le code de service et la chaîne de terminaison doivent utiliser des caractères minuscules. La chaîne a le format suivant : `YYYYMMDD/region/service/aws4_request`.
  - SigV4a : les informations d'identification comprennent la date au format `YYYYMMDD`, le nom du service et la chaîne de terminaison `aws4_request`, séparés par des barres obliques (/). Notez que la portée des informations d'identification n'inclut pas la région, car celle-ci est couverte dans un en-tête `X-Amz-Region-Set` distinct. La chaîne a le format suivant : `YYYYMMDD/service/aws4_request`.
- *HashedCanonicalRequest*— Le hash de la requête canonique, calculé à l'étape précédente.

L'exemple suivant est une chaîne de connexion.

```
"<Algorithm>" + "\n" +
timeStampISO8601Format + "\n" +
<Scope> + "\n" +
Hex(<Algorithm>(<CanonicalRequest>))
```

Dérivation d'une clé de signature

Pour dériver une clé de Signature, choisissez l'un des processus suivants afin de calculer une clé de Signature pour SigV4 ou SigV4a.

Dérivation d'une clé de Signature pour SigV4

Pour obtenir une clé de signature pour Sigv4, effectuez une succession d'opérations de hachage à clé (HMAC) à la date, à la région et au service de la demande, en utilisant votre clé d'accès AWS secrète comme clé pour l'opération de hachage initiale.

Pour chaque étape, appelez la fonction de hachage avec la clé et les données requises. Le résultat de chaque appel à la fonction de hachage devient l'entrée du prochain appel à la fonction de hachage.

L'exemple suivant montre comment vous obtenez la `SigningKey` utilisée dans la section suivante de cette procédure, en indiquant l'ordre dans lequel vos données sont concaténées et hachées. `HMAC-SHA256` est la fonction de hachage utilisée pour hacher les données comme indiqué.

```
DateKey = HMAC-SHA256("AWS4"+"<SecretAccessKey>", "<YYYYMMDD>")
DateRegionKey = HMAC-SHA256(<DateKey>, "<aws-region>")
DateRegionServiceKey = HMAC-SHA256(<DateRegionKey>, "<aws-service>")
SigningKey = HMAC-SHA256(<DateRegionServiceKey>, "aws4_request")
```

Entrée obligatoire

- `Key` : une chaîne contenant votre clé d'accès secrète.
- `Date` : une chaîne contenant la date utilisée dans l'étendue des informations d'identification, au format AAAAMMJJ
- `Region` : une chaîne contenant le code Région (par exemple, `us-east-1`).

  Pour obtenir la liste des chaînes régionales, consultez la section Points de terminaison régionaux dans les Références générales AWS.
- `Service` : une chaîne contenant le code de service (par exemple, `ec2`).
- Chaîne de connexion que vous avez créée à l'étape précédente.

Pour dériver une clé de Signature pour SigV4

1. Concaténez `"AWS4"` et la clé d'accès secrète. Appelez la fonction de hachage avec la chaîne concaténée en tant que clé et la chaîne de date en tant que données.

   ```
   DateKey = hash("AWS4" + Key, Date)
   ```
2. Appelez la fonction de hachage en utilisant le résultat de l'appel précédent en tant que clé et la chaîne Région en tant que données.

   ```
   DateRegionKey = hash(kDate, Region)
   ```
3. Appelez la fonction de hachage en utilisant le résultat de l'appel précédent en tant que clé et la chaîne de service en tant que données.

Le code de service est défini par le service. Vous pouvez utiliser get-products dans l'interface de ligne de commande de tarification AWS pour renvoyer le code de service d'un service.

```
DateRegionServiceKey = hash(kRegion, Service)
```

4. Appelez la fonction de hachage en utilisant le résultat de l'appel précédent en tant que clé et « aws4_request » en tant que données.

```
SigningKey = hash(kService, "aws4_request")
```

Dérivation d'une clé de Signature pour SigV4a

Pour créer une clé de Signature pour SigV4a, utilisez le processus suivant afin de dériver une paire de clés à partir de la clé d'accès secrète. Pour un exemple d'implémentation de cette dérivation, voir l'implémentation de l'authentification côté client par la AWS bibliothèque C99

```
n = [NIST P-256 elliptic curve group order]
G = [NIST P-256 elliptic curve base point]
label = "AWS4-ECDSA-P256-SHA256"

akid = [AWS access key ID as a UTF8 string]
sk = [AWS secret access Key as a UTF8 Base64 string]

input_key = "AWS4A" || sk
count = 1
while (counter != 255) {
  context = akid || counter // note: counter is one byte
  key = KDF(input_key, label, context, 256)
  c = Oct2Int(key)
  if (c > n - 2) {
    counter++
  } else {
    k = c + 1   // private key
    Q = k * G   // public key
  }
}

if (c < 255) {
  return [k, Q]
} else {
```

```
  return FAILURE
}
```

Calcul de la signature

Après avoir dérivé la clé de signature, calculez la signature à ajouter à votre demande. Cette procédure varie en fonction de la version de signature que vous utilisez.

Pour calculer une Signature SigV4

1. Appelez la fonction de hachage en utilisant le résultat de l'appel précédent en tant que clé et la chaîne de signature en tant que données. Utilisez la clé de signature dérivée comme la clé de hachage pour cette opération. Le résultat est la signature sous forme de valeur binaire.

   ```
   signature = hash(SigningKey, string-to-sign)
   ```

2. Convertissez la signature de la représentation binaire à la représentation hexadécimale, en caractères minuscules.

Pour calculer une Signature SigV4a

1. À l'aide de l'algorithme de signature numérique (ECDSA P-256), signez la chaîne à signer que vous avez créée à l'étape précédente. La clé utilisée pour cette signature est la clé asymétrique privée dérivée de la clé d'accès secrète telle que décrite ci-dessus.

   ```
   signature = base16(ECDSA-Sign(k, string-to-sign))
   ```

2. Convertissez la signature de la représentation binaire à la représentation hexadécimale, en caractères minuscules.

Ajout d'une signature à la requête

Ajoutez la signature calculée à votre demande.

Example Exemple : en-tête d'autorisation

SigV4

L'exemple suivant montre un `Authorization` en-tête pour l'`DescribeInstances`action utilisant AWS SIGv4. Pour des raisons de lisibilité, cet exemple est formaté avec des sauts de ligne.

Dans votre code, il doit s'agir d'une chaîne continue. Il n'y a pas de virgule entre l'algorithme et `Credential`. Toutefois, les autres éléments doivent être séparés par des virgules.

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request,
SignedHeaders=host;x-amz-date,
Signature=calculated-signature
```

SigV4a

L'exemple suivant montre un en-tête d'autorisation pour l'`CreateBucket`action utilisant AWS SigV4a. Pour des raisons de lisibilité, cet exemple est formaté avec des sauts de ligne. Dans votre code, il doit s'agir d'une chaîne continue. Il n'y a pas de virgule entre l'algorithme et Informations d'identification. Toutefois, les autres éléments doivent être séparés par des virgules.

```
Authorization: AWS4-ECDSA-P256-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20220830/s3/aws4_request,
SignedHeaders=host;x-amz-date;x-amz-region-set,
Signature=calculated-signature
```

Example Exemple : requête avec des paramètres d'authentification dans la chaîne de requête

SigV4

L'exemple suivant montre une requête pour l'`DescribeInstances`action utilisant AWS SIGv4 qui inclut les informations d'authentification. Pour des raisons de lisibilité, cet exemple est formaté avec des sauts de ligne et n'est pas code en URL. Dans votre code, la chaîne de requête doit être une chaîne continue codée en URL.

```
https://ec2.amazonaws.com/?
Action=DescribeInstances&
Version=2016-11-15&
X-Amz-Algorithm=AWS4-HMAC-SHA256&
X-Amz-Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request&
X-Amz-Date=20220830T123600Z&
X-Amz-SignedHeaders=host;x-amz-date&
X-Amz-Signature=calculated-signature
```

SigV4a

L'exemple suivant présente une requête pour l'action `CreateBucket` en utilisant AWS SigV4a qui inclut les informations d'authentification. Pour des raisons de lisibilité, cet exemple est formaté avec des sauts de ligne et n'est pas code en URL. Dans votre code, la chaîne de requête doit être une chaîne continue codée en URL.

```
https://ec2.amazonaws.com/?
Action=CreateBucket&
Version=2016-11-15&
X-Amz-Algorithm=AWS4-ECDSA-P256-SHA256&
X-Amz-Credential=AKIAIOSFODNN7EXAMPLE/20220830/s3/aws4_request&
X-Amz-Region-Set=us-west-1&
X-Amz-Date=20220830T123600Z&
X-Amz-SignedHeaders=host;x-amz-date;x-amz-region-set&
X-Amz-Signature=calculated-signature
```

Demander des exemples de signature

Les exemples de demandes de AWS signature suivants vous montrent comment utiliser SIGv4 pour signer des demandes envoyées sans le AWS SDK ou l'outil de ligne de AWS commande.

Chargement d'Amazon S3 basé sur le navigateur avec HTTP POST

Demandes d'authentification : Browser-Based les téléchargements décrivent la signature et les informations pertinentes qu'Amazon S3 utilise pour calculer la signature lors de la réception de la demande.

Exemple : Browser-Based Upload using HTTP POST (Using AWS Signature Version 4) fournit plus d'informations avec un exemple de politique POST et un formulaire que vous pouvez utiliser pour charger un fichier. L'exemple de politique et les informations d'identification fictives vous montre le flux de travail ainsi que la signature et le hachage de la politique qui en résultent.

Demandes authentifiées VPC Lattice

Exemples pour les demandes authentifiées par Signature Version 4 (SigV4) fournit des exemples en Python et en Java qui vous montrent comment signer les demandes avec et sans intercepteurs personnalisés.

Utilisation de Signature Version 4 avec Amazon Translate

La traduction en direct dans le métavers montre comment créer une application qui produit une solution de traduction en temps quasi réel. Cette solution de traduction vocale utilise AWS SIGv4 dans le codage des flux d'événements pour produire des transcriptions en temps réel.

Utilisation de Signature Version 4 avec Neptune

Exemple : connexion à Neptune à l'aide de Python avec la signature Signature Version 4 montre comment envoyer des demandes signées à Neptune à l'aide de Python. Cet exemple inclut des variations pour l'utilisation d'une clé d'accès ou d'informations d'identification temporaires.

Signature des demandes HTTP à Amazon Glacier

Exemple de calcul de signature pour l'API de streaming vous guide dans les détails de la création d'une signature pour Upload Archive (POST archive), l'une des deux API de streaming d'Amazon Glacier.

Envoi de demandes HTTP à Amazon SWF

Envoyez des demandes HTTP à Amazon SWF présente le contenu de l'en-tête d'une demande JSON envoyée à Amazon SWF.

Calcul de signature pour les API de streaming dans Amazon OpenSearch Service

La signature d'une demande de recherche Amazon OpenSearch Service avec le AWS SDK pour PHP version 3 inclut un exemple d'envoi de requêtes HTTP signées à Amazon OpenSearch Service.

Exemples de projets dans AWS référentiel d'échantillons

Les exemples de projets suivants montrent comment signer des demandes pour envoyer des requêtes d'API Rest à AWS des services utilisant des langages courants tels que Python Node.js, Java, C#, Go et Rust.

Projets de Signature Version 4a

Le projet sigv4-signing-examples fournit des exemples montrant comment signer des requêtes avec Sigv4 pour envoyer des requêtes d'API Rest à l' Services AWS aide de langages courants tels que Python, Java, C# Node.js, Go et Rust.

Le projet sigv4a-signing-examples fournit des exemples de signature de requêtes d'API multirégionales, par exemple des points d'accès dans Amazon S3. Multi-Region

Publier sur AWS IoT Core

Le code Python vers lequel publier à AWS IoT Core l'aide du protocole HTTPS fournit des conseils sur la façon de publier des messages à AWS IoT Core l'aide du protocole HTTPS et de l'authentification AWS Sigv4. Il possède deux implémentations de référence, l'une en Python et l'autre en NodeJs.

L'application .Net Framework sur laquelle publier à AWS IoT Core l'aide du protocole HTTPS fournit des conseils sur la façon de publier des messages à AWS IoT Core l'aide du protocole HTTPS et de l'authentification AWS SIGv4. Ce projet inclut également une implémentation correspondant à .NET Core.

Résoudre les problèmes liés à la signature de Signature Version 4 pour AWS Demandes d'API

> **Important**
>
> À moins que vous n'utilisiez les AWS SDK ou l'interface de ligne de commande, vous devez écrire du code pour calculer les signatures qui fournissent des informations d'authentification dans vos demandes. Le calcul de la signature Sigv4 peut être une tâche complexe, et nous vous recommandons d'utiliser les AWS SDK ou la CLI dans la mesure du possible.

Lorsque vous développez du code qui crée une demande signée, vous pouvez recevoir le protocole HTTP 403 `SignatureDoesNotMatch` de Services AWS. Ces erreurs signifient que la valeur de signature de votre requête HTTP AWS ne correspond pas à la signature Service AWS calculée. Les erreurs HTTP 401 `Unauthorized` sont renvoyées lorsque les autorisations ne permettent pas à l'appelant d'effectuer la requête.

Les demandes d'API peuvent renvoyer une erreur si :

- La demande d'API n'est pas signée et elle utilise l'authentification IAM.
- Les informations d'identification IAM utilisées pour signer la demande sont incorrectes ou ne sont pas autorisées à invoquer l'API.
- La signature de la demande d'API signée ne correspond pas à la signature calculée par le service AWS .

- L'en-tête de demande d'API est incorrect.

> **Note**
>
> Mettez à jour votre protocole de AWS signature de Signature version 2 (SIGv2) vers AWS Signature version 4 (SIGv4) avant d'explorer d'autres solutions aux erreurs. Les services, tels qu'Amazon S3, et les régions ne prennent plus en charge la signature SigV2.

Causes possibles :

- Erreurs d'informations d'identification
- Erreurs de requête canonique et de chaîne de signature
- Erreurs d'étendue des informations d'identification
- Erreurs de signature de clé

Erreurs d'informations d'identification

Assurez-vous que la demande d'API est signée avec SigV4. Si la demande d'API n'est pas signée, il se peut que vous receviez le message d'erreur : `Missing Authentication Token`. Ajoutez la signature manquante et renvoyez la demande.

Vérifiez que les informations d'identification de la clé d'accès et de la clé secrète sont correctes. Si la clé d'accès est incorrecte, le message d'erreur suivant peut s'afficher : `Unauthorized`. Assurez-vous que l'entité utilisée pour signer la demande est autorisée à effectuer cette dernière. Pour en savoir plus, consultez Résolution des problèmes liés aux messages d'erreur d'accès rejeté.

Erreurs de requête canonique et de chaîne de signature

Si le calcul de la requête canonique est incorrect dans Création d'un hachage de la requête canonique ou Création d'une chaîne à signer, l'étape de vérification de la signature effectuée par le service échoue avec le message d'erreur :

```
The request signature we calculated does not match the signature you provided
```

Lorsque le AWS service reçoit une demande signée, il recalcule la signature. S'il existe des différences entre les valeurs, les signatures ne correspondent pas. Comparez la requête canonique

et la chaîne à votre demande signée avec la valeur du message d'erreur. Modifiez le processus de signature en cas de différence.

> **Note**
>
> Vous pouvez également vérifier que vous n'avez pas envoyé la requête via un proxy qui modifie les en-têtes ou la requête.

Example Exemple de requête canonique

```
GET                                                       -------- HTTP method
/                                                         -------- Path. For API stage
 endpoint, it should be /{stage-name}/{resource-path}
                                                          -------- Query string key-
value pair. Leave it blank if the request doesn't have a query string.
content-type:application/json                             -------- Header key-value
 pair. One header per line.
host:0123456789.execute-api.us-east-1.amazonaws.com       -------- Host and x-amz-date
 are required headers for all signed requests.
x-amz-date:20220806T024003Z

content-type;host;x-amz-date                              -------- A list of signed
 headers
d167e99c53f15b0c105101d468ae35a3dc9187839ca081095e340f3649a04501        -------- Hash
 of the payload
```

Pour vérifier que la clé secrète correspond à l'ID de la clé d'accès, vous pouvez les tester avec une implémentation fonctionnelle connue. Par exemple, utilisez un AWS SDK ou l' AWS interface de ligne de commande pour adresser une demande à AWS.

En-tête de la demande d'API

Lorsque l'en-tête d'autorisation est vide, que la clé ou la signature des informations d'identification est manquante ou incorrecte, que l'en-tête ne commence pas par un nom d'algorithme ou que les paires clé-valeur n'incluent pas de signe égal, vous recevez l'une des erreurs suivantes :

- L'en-tête d'autorisation ne peut pas être vide.
- L'en-tête d'autorisation nécessite le paramètre « Informations d'identification ».
- L'en-tête d'autorisation nécessite le paramètre « Signature ».

- La signature contient une paire clé=valeur non valide (signe égal manquant) dans l'en-tête d'autorisation.

Assurez-vous que l'en-tête d'autorisation SigV4 que vous avez ajouté dans Calcul de la signature comprend la clé d'informations d'identification correcte, ainsi que la date de la requête à l'aide de HTTP Date ou de l'en-tête `x-amz-date`.

Si vous avez reçu une IncompleteSignatureException erreur et que la construction de la signature est correcte, vous pouvez vérifier que l'en-tête d'autorisation n'a pas été modifié pendant le transfert vers le en Service AWS calculant un SHA-256 hachage et un codage B64 de l'en-tête d'autorisation dans votre demande côté client.

1. Obtenez l'en-tête d'autorisation que vous avez envoyé dans la requête. Votre en-tête d'autorisation ressemble à l'exemple suivant :

   ```
   Authorization: AWS4-HMAC-SHA256
   Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
   SignedHeaders=host;range;x-amz-date,
   Signature=example-generated-signature
   ```

2. Calculez un SHA-256 hachage de l'en-tête d'autorisation.

   ```
   hashSHA256(rawAuthorizationHeader) = hashedAuthorizationHeader
   ```

3. Codez l'en-tête d'autorisation haché au format Base64.

   ```
   base64(hashedAuthorizationHeader) = encodedHashedAuthorizationHeader
   ```

4. Comparez la chaîne hachée et codée que vous venez de calculer à la chaîne que vous avez reçue dans votre message d'erreur. Votre message d'erreur doit ressembler à l'exemple suivant :

   ```
   com.amazon.coral.service#IncompleteSignatureException:
   The signature contains an in-valid key=value pair (missing equal-sign)
   in Authorization header (hashed with SHA-256 and encoded with Base64):
   '9c574f83b4b950926da4a99c2b43418b3db8d97d571b5e18dd0e4f3c3ed1ed2c'.
   ```

- Si les deux hachages diffèrent, c'est qu'une partie de l'en-tête d'autorisation a été modifiée en transit Ce changement peut être dû au fait que les gestionnaires de votre réseau ou de votre client attachent des en-têtes signés ou modifient l'en-tête d'autorisation d'une manière ou d'une autre.

- Si les deux hachages correspondent, l'en-tête d'autorisation que vous avez envoyé dans la demande correspond à celui AWS reçu. Consultez le message d'erreur que vous avez reçu pour déterminer si le problème est dû à des informations d'identification ou à une signature incorrectes. Ces erreurs sont traitées dans les autres sections de cette page.

Erreurs d'étendue des informations d'identification

Le champ d'application des informations d'identification dans Création d'une chaîne à signer limite une signature à une date, une région et un service spécifiques. Cette chaîne présente le format suivant :

```
YYYYMMDD/region/service/aws4_request
```

> **Note**
>
> Si vous utilisez SigV4a, la région n'est pas incluse dans le champ d'application des informations d'identification.

Date

Si l'étendue des informations d'identification ne spécifie pas la même date que l'en-tête x-amz-date, l'étape de vérification de la signature échoue avec le message d'erreur suivant :

```
Date in Credential scope does not match YYYYMMDD from ISO-8601 version of date from
 HTTP
```

Si la requête indique une date future, l'étape de vérification de la signature échoue avec le message d'erreur suivant :

```
Signature not yet current: date is still later than date
```

Si la requête a expiré, l'étape de vérification de la signature échoue avec le message d'erreur suivant :

```
Signature expired: date is now earlier than date
```

Région

Si l'étendue des informations d'identification ne spécifie pas la même Région que la requête, l'étape de vérification de la signature échoue avec le message d'erreur suivant :

```
Credential should be scoped to a valid Region, not region-code
```

Service

Si l'étendue des informations d'identification ne spécifie pas le même service que l'en-tête host, l'étape de vérification de la signature échoue avec le message d'erreur suivant :

```
Credential should be scoped to correct service: 'service'
```

Chaîne de terminaison

Si l'étendue des informations d'identification ne se termine pas par aws4_request, l'étape de vérification de la signature échoue avec le message d'erreur suivant :

```
Credential should be scoped with a valid terminator: 'aws4_request'
```

Erreurs de signature de clé

Les erreurs causées par une dérivation incorrecte de la clé de signature ou une utilisation inappropriée du chiffrement sont plus difficiles à résoudre. Après avoir vérifié que la chaîne canonique et la chaîne à signer sont correctes, vous pouvez également vérifier l'existence de l'un des problèmes suivants :

- La clé d'accès secrète ne correspond pas à l'ID de clé d'accès que vous avez spécifiée.
- Votre code de dérivation de clé pose problème.

Pour vérifier que la clé secrète correspond à l'ID de la clé d'accès, vous pouvez les tester avec une implémentation fonctionnelle connue. Par exemple, utilisez un AWS SDK ou le AWS CLI pour faire une demande à AWS. Pour obtenir des exemples, consultez Demander des exemples de signature

Référence de politique JSON IAM

Cette section présente la syntaxe, les descriptions et des exemples détaillés des éléments, des variables et de la logique d'évaluation des politiques JSON dans IAM. Pour plus d'informations générales, consultez Présentation des politiques JSON.

Cette référence comprend les sections suivantes.

- Référence de l'élément de politique JSON IAM : en savoir plus sur les éléments que vous pouvez utiliser lorsque vous créez une politique. Consultez des exemples de politiques supplémentaires, et découvrez les conditions et les types de données pris en charge, ainsi que la manière dont ils sont utilisés dans différents services.
- Logique d'évaluation de politiques— Cette section décrit les AWS demandes, la manière dont elles sont authentifiées et la manière dont les politiques sont AWS utilisées pour déterminer l'accès aux ressources.
- Syntaxe du langage de politique JSON IAM – Cette section présente une syntaxe formelle du langage de stratégie utilisé pour créer des politiques dans IAM.
- AWS politiques gérées pour les fonctions professionnelles – Cette section répertorie toutes les politiques gérées par AWS qui correspondent directement à des fonctions professionnelles courantes du secteur de l'informatique. Utilisez ces politiques pour accorder les autorisations nécessaires à l'exécution des tâches prévues pour quelqu'un occupant une fonction professionnelle spécifique. Ces politiques consolident les autorisations pour de nombreux services en une politique unique.
- AWS clés de contexte de condition globale— Cette section inclut une liste de toutes les clés de condition AWS globales que vous pouvez utiliser pour limiter les autorisations dans une politique IAM.
- IAM et AWS STS clés de contexte de condition— Cette section inclut une liste de toutes les clés IAM et de AWS STS condition que vous pouvez utiliser pour limiter les autorisations dans le cadre d'une politique IAM.
- Actions, ressources et clés de condition pour les AWS services : cette section présente une liste de toutes les opérations d' AWS API que vous pouvez utiliser comme autorisations dans une politique IAM. Elle inclut également les clés de condition spécifiques au service qui peuvent être utilisées pour affiner davantage la demande.

Référence de l'élément de politique JSON IAM

Les documents de politique JSON se composent d'éléments. Les éléments sont répertoriés dans l'ordre général d'utilisation dans une politique. L'ordre des éléments n'a pas d'importance. Par exemple, l'élément `Resource` peut venir avant l'élément `Action`. Il n'est pas nécessaire de spécifier d'éléments `Condition` dans une politique. Pour en savoir plus sur la structure générale et la fonction d'un document de politique JSON, consultez Présentation des politiques JSON.

Certains éléments de politique JSON s'excluent mutuellement. Cela signifie que vous ne pouvez pas créer une politique qui les utilise ensemble. Par exemple, vous ne pouvez pas utiliser `Action` et `NotAction` dans la même instruction de politique. Les autres paires qui s'excluent mutuellement sont notamment `Principal`/`NotPrincipal` et `Resource`/`NotResource`.

Les détails inclus dans une politique varient pour chaque service, selon les actions autorisées par le service, les types de ressources qu'il contient, etc. Lorsque vous créez des politiques pour un service spécifique, il peut être utile de consulter des exemples de stratégie pour ce service. Pour consulter la liste de tous les services prenant en charge IAM et pour obtenir des liens vers la documentation de ces services relative à IAM et aux politiques, consultez AWS services qui fonctionnent avec IAM.

Lorsque vous créez ou modifiez une politique JSON, IAM peut effectuer une validation de politique pour vous aider à créer une politique efficace. IAM identifie les erreurs de syntaxe JSON, tandis que IAM Access Analyzer fournit des vérifications de politique supplémentaires avec des recommandations pour vous aider à affiner vos politiques. Pour en savoir plus sur la validation de politiques, veuillez consulter Validation de politique IAM. Pour en savoir plus sur les vérifications des politiques IAM Access Analyzer et les recommandations exploitables, veuillez consulter Validation de politique IAM Access Analyzer.

Rubriques

- Éléments de politique JSON IAM : Version
- Éléments de politique JSON IAM : Id
- Éléments de politique JSON IAM : Statement
- Éléments de politique JSON IAM : Sid
- Éléments de la politique JSON IAM : Effet
- AWS Éléments de politique JSON : Principal
- AWS Éléments de politique JSON : NotPrincipal
- Éléments de la politique JSON IAM : Action
- Éléments de politique JSON IAM : NotAction
- Éléments de la politique JSON IAM : Ressource
- Éléments de la politique JSON IAM : NotResource
- Éléments de la politique JSON IAM : Condition
- Éléments des politiques IAM : variables et balises
- Éléments de politique JSON IAM : Types de données pris en charge

Éléments de politique JSON IAM : Version

> **Remarque pour lever l'ambiguïté**
>
> Cet élément de politique JSON `Version` ne représente pas la même chose qu'une version de politique. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. En revanche, une version de politique est créée lorsque vous apportez des modifications à une politique gérée par le client dans IAM. La politique modifiée ne remplace pas la politique existante. À la place, IAM crée une nouvelle version de la politique gérée. Si vous recherchez des informations sur les différentes versions prises en charge qui sont disponibles pour les politiques gérées, consultez the section called "Gestion des versions des politiques IAM".

L'élément de politique `Version` spécifie les règles de syntaxe de langage qui doivent être utilisées pour traiter une politique. Pour utiliser toutes les fonctions de politique disponibles, incluez l'élément `Version` suivant à l'extérieur de l'élément `Statement` dans toutes vos politiques.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

IAM prend en charge les valeurs d'élément `Version` suivantes :

- `2012-10-17`. Il s'agit de la version actuelle du langage de politique ; vous devez toujours inclure un élément `Version` et définir sa valeur sur `2012-10-17`. Sinon, vous ne pouvez pas utiliser des fonctions telles que les variables de politique qui ont été introduites avec cette version.
- `2008-10-17`. Ceci est une version antérieure du langage de politique. Elle peut figurer dans des politiques existantes antérieures. N'utilisez pas cette version pour les nouvelles politiques ou

lors de la mise à jour de politiques existantes. Les nouvelles fonctions, telles que les variables de politique, ne fonctionneront pas avec votre politique. Par exemple, les variables telles que `${aws:username}` ne sont pas reconnues et sont traitées comme des chaînes littérales dans la politique.

Éléments de politique JSON IAM : Id

L'élément `Id` spécifie un identifiant facultatif pour la politique. L'ID est utilisé de différentes façons, selon les services. L'ID est autorisé dans les politiques basées sur les ressources, mais pas dans celles basées sur une identité.

Dans le cas de services vous permettant de définir un élément `ID`, il est recommandé d'utiliser un UUID (GUID) comme valeur, ou d'incorporer un UUID dans l'ID pour en garantir l'unicité.

JSON

```
{
  "Version":"2012-10-17",
  "Id": "cd3ad3d9-2776-4ef1-a904-4c229d1642ee",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Certains AWS services (par exemple, Amazon SQS ou Amazon SNS) peuvent nécessiter cet élément et avoir des exigences d'unicité pour celui-ci. Pour obtenir des informations spécifiques à chaque service sur la création de politiques, reportez-vous à la documentation du service que vous utilisez.

Éléments de politique JSON IAM : Statement

L'élément `Statement` constitue l'élément principal d'une politique. Cet élément est obligatoire. L'élément `Statement` peut contenir une seule instruction ou un tableau d'instructions. Chaque bloc d'instruction doit être placé entre accolades { }. Pour plusieurs instructions, le tableau doit être placé entre crochets [ ].

```
"Statement": [{...},{...},{...}]
```

L'exemple suivant illustre une politique qui contient un tableau de trois instructions dans un même élément `Statement`. (La politique vous permet d'accéder à votre propre « dossier principal » dans la console Amazon S3.) La politique inclut la variable `aws:username`, qui est remplacée par le nom utilisateur provenant de la demande lors de l'évaluation de la politique. Pour de plus amples informations, veuillez consulter Introduction.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/"
      ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
```

```
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}",
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}/*"
      ]
    }
  ]
}
```

Éléments de politique JSON IAM : Sid

Vous pouvez fournir un `Sid` (ID d'instruction) en tant qu'identifiant facultatif pour l'instruction de politique. Vous pouvez affecter une valeur `Sid` à chaque instruction d'un tableau de instructions. Vous pouvez utiliser la valeur `Sid` comme description de l'instruction de politique. Dans un service qui vous permet de spécifier un élément `ID` tel que SQS et SNS, la valeur `Sid` est un simple sous-identifiant de l'ID du document de politique. Dans IAM, la valeur `Sid` doit être unique dans une politique JSON.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExampleStatementID",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

L'élément `Sid` prend en charge les majuscules ASCII (A à Z), les minuscules (a à z) et les chiffres (0 à 9).

IAM n'expose pas le `Sid` dans l'API IAM. Il n'est pas possible d'extraire une instruction spécifique en vous basant sur cet ID.

> **Note**
>
> Certains AWS services (par exemple, Amazon SQS ou Amazon SNS) peuvent nécessiter cet élément et avoir des exigences d'unicité pour celui-ci. Pour obtenir des informations spécifiques à chaque service sur la création de politiques, reportez-vous à la documentation du service que vous utilisez.

Éléments de la politique JSON IAM : Effet

L'élément `Effect` est requis ; il spécifie si l'instruction génère une autorisation ou un refus explicite. Les valeurs valides pour `Effect` sont `Allow` et `Deny`. La valeur `Effect` est sensible à la casse.

```
"Effect":"Allow"
```

Par défaut, l'accès aux ressources est refusé. Pour autoriser l'accès à une ressource, vous devez définir l'élément `Effect` sur `Allow`. Pour remplacer une autorisation (par exemple, pour remplacer une autorisation qui autrement est en vigueur), vous définissez l'élément `Effect` sur `Deny`. Pour de plus amples informations, veuillez consulter Logique d'évaluation de politiques.

AWS Éléments de politique JSON : Principal

Utilisez l'élément `Principal` dans une politique JSON basée sur les ressources pour spécifier le principal qui est autorisé ou non à accéder à une ressource.

Vous devez utiliser l'élément `Principal` dans les politiques basées sur les ressources. Plusieurs services prennent en charge les politiques basées sur les ressources, y compris IAM. Le type de politique basée sur les ressources IAM est une politique d'approbation de rôle. Dans les rôles IAM, utilisez l'élément `Principal` dans la politique d'approbation du rôle pour spécifier qui peut endosser le rôle. Lorsqu'il s'agit d'un accès entre comptes, vous devez spécifier l'identifiant à 12 chiffres du compte approuvé. Pour savoir si les principaux des comptes situés en dehors de votre zone de confiance (organisation ou compte de confiance) ont accès à vos rôles, consultez Qu'est-ce que l'Analyseur d'accès IAM ?.

> **Note**
>
> Après avoir créé le rôle, vous pouvez remplacer le compte par « * » pour autoriser tout le monde à endosser le rôle. Dans ce cas, nous vous recommandons vivement de limiter les personnes autorisées à accéder au rôle d'une autre manière, par exemple avec un élément

`Condition` qui limite l'accès à certaines adresses IP. Votre rôle ne doit pas être accessible à n'importe qui !

D'autres exemples de ressources prenant en charge les politiques basées sur les ressources incluent un compartiment Amazon S3 ou une interface AWS KMS key.

Vous ne pouvez pas utiliser `Principal` cet élément dans une politique basée sur l'identité. Identity-based les politiques sont des politiques d'autorisations que vous associez aux identités IAM (utilisateurs, groupes ou rôles). Dans ces cas, le principal est implicitement l'identité à laquelle est attachée la politique.

Rubriques

- Comment spécifier un principal
- Compte AWS principals
- Principaux de rôles IAM
- Principaux de séance de rôle
- Principaux fédérés OIDC
- Principaux fédérés SAML
- Principaux de l'utilisateur IAM
- Principaux d'IAM Identity Center
- AWS STS principes d'utilisateur fédérés
- AWS principes de service
- AWS responsables de service dans les régions optionnelles
- Tous les principaux
- En savoir plus

Comment spécifier un principal

Vous spécifiez un principal dans l'élément `Principal` d'une politique basée sur les ressources ou des clés de condition prenant en charge les principaux.

Vous pouvez spécifier l'un des principaux suivants dans une politique :

- Compte AWS et utilisateur root

- Rôles IAM
- Séances de rôle
- Utilisateurs IAM
- principaux d'utilisateur fédéré
- AWS services
- Tous les principaux

Vous ne pouvez pas identifier un groupe d'utilisateurs en tant que principal dans une politique (telle qu'une politique basée sur les ressources), car les groupes concernent les autorisations, et non l'authentification, et les principaux sont des entités IAM authentifiées.

Vous pouvez spécifier plus d'un principal pour chacun des types de principaux dans les sections suivantes, à l'aide d'un tableau. Les tableaux peuvent inclure une ou plusieurs valeurs. Lorsque vous spécifiez plus d'un principal dans un élément, vous accordez des autorisations à chaque principal. Ceci est un `OR` logique et non un `AND` logique, car vous êtes authentifié comme un unique principal à la fois. Si vous incluez plus d'une valeur, utilisez des crochets (`[` et `]`) et délimitez chaque entrée du tableau par une virgule. L'exemple de politique suivant définit les autorisations pour le compte 123456789012 ou le compte 555555555555.

```
"Principal" : {
"AWS": [
  "123456789012",
  "555555555555"
  ]
}
```

> **Note**
>
> Vous ne pouvez pas utiliser un caractère générique pour faire correspondre une partie d'un nom de principal ou d'un ARN.

Compte AWS principals

Vous pouvez spécifier Compte AWS des identifiants dans l'`Principal`élément d'une politique basée sur les ressources ou dans des clés de condition qui prennent en charge les principes. Cela délègue l'autorité sur le compte. Lorsque vous autorisez l'accès à un autre compte, un administrateur de ce

compte doit alors accorder l'accès à une identité (utilisateur ou rôle IAM) de ce compte. Lorsque vous spécifiez un Compte AWS, vous pouvez utiliser l'ARN du compte (arn:aws:iam : ::root*account-ID*) ou une forme abrégée composée du préfixe suivi de l'identifiant du compte. `"AWS":`

Par exemple, soit un ID de compte `123456789012`, vous pouvez utiliser l'une des méthodes suivantes pour spécifier ce compte dans l'élément `Principal` :

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

```
"Principal": { "AWS": "123456789012" }
```

L'ARN du compte et l'ID de compte abrégé se comportent de la même manière. Les deux délèguent des autorisations au compte. L'utilisation de l'ARN du compte dans l'élément `Principal` ne limite pas les autorisations au seul utilisateur racine du compte.

> **Note**
>
> Lorsque vous enregistrez une politique basée sur les ressources qui inclut l'ID du compte abrégé, le service peut le convertir en ARN principal. Cela ne modifie pas la fonctionnalité de la politique.

Certains AWS services proposent des options supplémentaires pour spécifier le principal d'un compte. Par exemple, Amazon S3 vous permet de spécifier un ID d'utilisateur canonique à l'aide du format suivant :

```
"Principal": { "CanonicalUser":
 "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be" }
```

Vous pouvez également en spécifier plusieurs Compte AWS(ou ID utilisateur canonique) en tant que principal à l'aide d'un tableau. Par exemple, vous pouvez spécifier un principal dans une politique de compartiment à l'aide des trois méthodes.

```
"Principal": {
  "AWS": [
    "arn:aws:iam::123456789012:root",
    "999999999999"
  ],
```

```
  "CanonicalUser": "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be"
}
```

Principaux de rôles IAM

Vous pouvez spécifier les ARN principaux du rôle IAM dans l'élément `Principal` d'une politique basée sur les ressources ou des clés de condition prenant en charge les principaux. Les rôles IAM sont des identités. Dans IAM, les identités sont des ressources auxquelles vous pouvez attribuer des autorisations. Les rôles font confiance à une autre identité authentifiée pour endosser ce rôle. Il s'agit notamment d'un principal dans l'interface AWS ou d'un utilisateur provenant d'un fournisseur d'identité externe (IdP). Lorsqu'un principal ou une identité endosse un rôle, ils reçoivent des informations d'identification de sécurité temporaires avec les autorisations du rôle endossé. Lorsqu'ils utilisent ces informations d'identification de session pour effectuer des opérations AWS, ils deviennent un rôle principal de session.

Lorsque vous spécifiez un principal de rôle dans une politique basée sur les ressources, les autorisations effectives du principal sont limitées par tous les types de politique limitant les autorisations pour le rôle. Cela inclut les politiques de séance et les limites d'autorisations. Pour plus d'informations sur l'évaluation des autorisations effectives pour une séance de rôle, veuillez consulter Logique d'évaluation de politiques.

Pour spécifier l'ARN du rôle dans l'élément `Principal`, utilisez le format suivant :

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:role/role-name" }
```

> **Important**
>
> Si votre élément `Principal` dans une politique d'approbation de rôle contient un ARN qui pointe vers un rôle IAM spécifique, alors cet ARN se transforme en l'ID principal unique du rôle lorsque vous enregistrez la politique. Cela permet de réduire le risque d'escalade des privilèges par la suppression et la nouvelle création du rôle. Normalement, vous ne voyez pas cet ID dans la console, car IAM utilise une transformation inverse pour revenir au rôle ARN lorsque la politique d'approbation est affichée.
> Cependant, si vous supprimez le rôle, vous interrompez la relation. La politique ne s'applique plus, même si vous recréez le rôle, étant donné que le nouvel ID du principal du nouveau rôle ne correspond pas à l'ID stocké dans la politique d'approbation. Dans ce cas, l'ID principal apparaît dans les politiques basées sur les ressources car il n'est plus AWS possible de le mapper à un ARN valide.

Par conséquent, si vous supprimez et recréez un rôle référencé dans l'`Principal`élément d'une politique de confiance, vous devez modifier le rôle dans la politique pour remplacer l'ID principal par l'ARN correct. L'ARN se transforme à nouveau en nouvel ID principal du rôle lorsque vous enregistrez la politique. Pour plus d'informations, consultez la section Comprendre AWS la gestion des rôles IAM supprimés dans les politiques.

Vous pouvez également spécifier le principal du rôle comme principal dans une politique basée sur les ressources ou créer une politique d'autorisation étendue qui utilise la clé de condition `aws:PrincipalArn`. Lorsque vous utilisez cette clé, le principal de séance de rôle reçoit les autorisations en fonction de l'ARN du rôle qui a été endossé, et non de l'ARN de la séance résultante. Comme il AWS ne convertit pas les ARN des clés de condition en ID, les autorisations accordées à l'ARN du rôle sont conservées si vous supprimez le rôle, puis créez un nouveau rôle portant le même nom. Identity-based les types de politiques, tels que les limites d'autorisations ou les politiques de session, ne limitent pas les autorisations accordées à l'aide de la clé de `aws:PrincipalArn` condition avec un caractère générique (*) dans l'`Principal`élément, sauf si les politiques basées sur l'identité contiennent un refus explicite.

Principaux de séance de rôle

Vous pouvez spécifier des séances de rôle dans l'élément `Principal`d'une politique basée sur les ressources ou des clés de condition prenant en charge les principaux. Lorsqu'un principal ou une identité endosse un rôle, ils reçoivent des informations d'identification de sécurité temporaires avec les autorisations du rôle endossé. Lorsqu'ils utilisent ces informations d'identification de session pour effectuer des opérations AWS, ils deviennent un rôle principal de session.

Le format que vous utilisez pour un rôle principal de session dépend de l' AWS STS opération qui a été utilisée pour assumer le rôle.

Important

AWS recommande d'utiliser les principaux de rôle IAM dans vos politiques plutôt que les principaux de session de rôle dans la mesure du possible. Utilisez `Condition` des instructions et des clés de condition pour restreindre davantage l'accès si nécessaire.

Pour spécifier l'ARN principal de la session de rôle dans l'`Principal`élément, utilisez le format suivant :

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:assumed-role/role-name/role-session-name" }
```

En outre, les administrateurs peuvent concevoir un processus pour contrôler la façon dont les séances de rôle sont émises. Par exemple, ils peuvent fournir une solution en un clic à leurs utilisateurs qui crée un nom de séance prévisible. Si votre administrateur procède ainsi, vous pouvez utiliser des principaux de séance de rôle dans vos politiques ou clés de condition. Sinon, vous pouvez spécifier l'ARN du rôle comme principal dans la `aws:PrincipalArn` clé de condition. La façon dont vous spécifiez le rôle comme principal peut modifier les autorisations effectives pour la séance résultante. Pour de plus amples informations, veuillez consulter Principaux de rôles IAM.

Principaux fédérés OIDC

Un principal fédéré OIDC est le principal utilisé lors de l'appel d'une AWS STS `AssumeRoleWithWebIdentity` API avec un jeton Web JSON (JWT) à partir d'un IDP conforme à l'OIDC, également appelé OpenID Provider (OP), pour demander des informations d'identification temporaires. AWS Un principal fédéré OIDC peut représenter un IDP OIDC sur votre AWS compte ou les 4 fournisseurs d'identité intégrés :Login with Amazon, GoogleFacebook, et Amazon Cognito. https://docs.aws.amazon.com/cognito/latest/developerguide/role-based-access-control.html

Les utilisateurs, les charges de travail ou les systèmes auxquels un JWT a été attribué par leur IDP OIDC peuvent appeler à `AssumeRoleWithWebIdentity` l'aide du JWT pour demander des informations d'identification de AWS sécurité temporaires pour un rôle IAM configuré pour faire confiance à l'IDP OIDC qui a émis le JWT. Le JWT peut être un jeton d'identification, un jeton d'accès ou un jeton JWT délivré par toute autre méthode, à condition qu'il réponde aux exigences énumérées par AWS STS. Pour plus d'informations, consultez les Scénarios courants et Demande d'informations d'identification par le biais d'un fournisseur OIDC.

Utilisez ce type de principal dans votre politique de confiance des rôles pour autoriser ou refuser les autorisations d'appel à `AssumeRoleWIthWebIdentity` l'aide d'un IDP OIDC qui existe dans votre Compte AWS ou de l'un des quatre IDP intégrés. Pour spécifier l'ARN du principal fédéré OIDC dans l'élément `Principal` d'une politique de confiance de rôle, utilisez l'un des quatre formats suivants pour les IDP OIDC intégrés :

```
"Principal": { "Federated": "cognito-identity.amazonaws.com" }
```

```
"Principal": { "Federated": "www.amazon.com" }
```

```
"Principal": { "Federated": "graph.facebook.com" }
```

```
"Principal": { "Federated": "accounts.google.com" }
```

Lorsque vous utilisez un fournisseur OIDC que vous ajoutez à votre compte, par exemple GitHub, vous spécifiez l'ARN du fournisseur dans la politique de confiance de votre rôle. Cette configuration vous permet de rédiger des politiques IAM qui contrôlent spécifiquement l'accès des utilisateurs authentifiés via votre fournisseur d'identité personnalisé.

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:oidc-provider/full-OIDC-identity-provider-URL" }
```

Par exemple, si GitHub était le fournisseur d'identité Web de confiance, l'ARN de session de rôle OIDC dans l'élément `Principal` d'une politique de confiance de rôle utilise le format suivant :

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:oidc-provider/
tokens.actions.githubusercontent.com" }
```

Pour plus d'informations, consultez Configuration d'OpenID Connect dans Amazon Web Services.

Les principaux fédérés OIDC ne sont pas pris en charge dans les types de politiques autres que les politiques de confiance des rôles.

Principaux fédérés SAML

Un principal fédéré SAML est un principal utilisé lors de l'appel d'une AWS STS `AssumeRoleWithSAML` API pour demander des informations d' AWS identification temporaires à l'aide d'une assertion SAML. Vous pouvez utiliser votre fournisseur d'identité (IdP) SAML externe (IdP) pour vous connecter, puis endosser un rôle IAM à l'aide de cette opération. Semblable à`AssumeRoleWithWebIdentity`, `AssumeRoleWithSAML` ne nécessite pas AWS d'informations d'identification pour l'authentification. Au lieu de cela, les utilisateurs s'authentifient d'abord auprès de leur fournisseur d'identité SAML, puis effectuent l'appel d'`AssumeRoleWithSAML`API à l'aide de leur assertion SAML, ou sont redirigés vers la AWS Sign-In/SAML page pour se connecter à. Console de gestion AWS Pour de plus amples informations sur les principaux pouvant endosser un rôle à l'aide de cette opération, veuillez consulter Comparez les AWS STS informations d'identification.

Utilisez ce type de principal dans votre politique de confiance des rôles pour autoriser ou rejeter des autorisations en fonction du fournisseur d'identité SAML approuvé. Pour spécifier l'ARN de séance

du rôle d'identité SAML dans l'élément `Principal` d'une politique d'approbation de rôle, utilisez le format suivant :

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:saml-provider/provider-name" }
```

Principaux de l'utilisateur IAM

Vous pouvez spécifier des utilisateurs IAM dans l'élément `Principal` d'une politique basée sur les ressources ou dans les clés de condition qui prennent en charge les principes.

> **Note**
>
> Dans un élément `Principal`, le nom utilisateur faisant partie d'Amazon Resource Name (ARN) est sensible à la casse.

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:user/user-name" }
```

```
"Principal": {
  "AWS": [
    "arn:aws:iam::AWS-account-ID:user/user-name-1",
    "arn:aws:iam::AWS-account-ID:user/user-name-2"
  ]
}
```

Lors de la spécification d'utilisateurs dans un élément `Principal`, il n'est pas possible d'utiliser un caractère générique (*) signifiant « tous les utilisateurs ». Les principaux doivent toujours nommer des utilisateurs spécifiques.

> **Important**
>
> Si votre élément `Principal` dans une politique d'approbation de rôle comporte un ARN qui pointe vers un utilisateur IAM spécifique, IAM transforme l'ARN en ID du principal unique de l'utilisateur lorsque vous enregistrez la politique. Cela permet de réduire le risque d'escalade des privilèges par la suppression et la nouvelle création de l'utilisateur. Cet ID n'est pas fréquent dans la console, car il existe également une transformation inverse, pour revenir à l'ARN de l'utilisateur, lorsque la politique d'approbation est affichée.
> Cependant, si vous supprimez l'utilisateur, vous interrompez la relation La politique ne s'applique plus, même si vous recréez l'utilisateur. Cela est dû au fait que le nouvel ID du

principal du nouvel utilisateur ne correspond pas à l'ID stocké dans la politique d'approbation. Dans ce cas, l'ID principal apparaît dans les politiques basées sur les ressources car il n'est plus AWS possible de le mapper à un ARN valide.
Par conséquent, si vous supprimez et recréez un utilisateur référencé dans l'élément `Principal` d'une politique d'approbation, vous devez modifier le rôle afin de remplacer l'ID du principal qui est désormais incorrect par l'ARN correct. IAM transforme à nouveau l'ARN en le nouvel ID du principal de l'utilisateur lorsque vous enregistrez la politique.

Principaux d'IAM Identity Center

Dans IAM Identity Center, le principal d'une politique basée sur les ressources doit être défini comme étant le principal Compte AWS . Pour spécifier l'accès, faites référence à l'ARN du rôle du jeu d'autorisations dans le bloc de conditions. Pour en savoir, consultez Référencement des jeux d'autorisations dans les politiques de ressources, Amazon EKS et AWS KMS (français non garanti) dans le Guide de l'utilisateur IAM Identity Center.

AWS STS principes d'utilisateur fédérés

Vous pouvez spécifier des séances d'utilisateur fédéré dans l'élément `Principal` d'une politique basée sur les ressources ou dans les clés de condition qui prennent en charge les principaux.

⚠ Important

AWS vous recommande de limiter l'utilisation des sessions utilisateur AWS STS fédérées. À la place, utilisez des rôles IAM.

Un utilisateur principal AWS STS fédéré est créé par le biais de l'`GetFederationToken`opération appelée avec des informations d'identification IAM de longue durée. Les autorisations utilisateur fédérées correspondent à l'intersection entre le principal qui a appelé `GetFederationToken` et les politiques de session transmises en tant que paramètres à l'API `GetFederationToken`.

Dans AWS, les utilisateurs IAM Utilisateur racine d'un compte AWS peuvent s'authentifier à l'aide de clés d'accès à long terme. Pour plus d'informations sur les principaux qui peuvent se fédérer à l'aide de cette opération, veuillez consulter Comparez les AWS STS informations d'identification.

- Utilisateur fédéré IAM : un utilisateur IAM se fédère à l'aide de l'opération `GetFederationToken` qui donne lieu à une session d'utilisateur fédéré pour cet utilisateur IAM.

- Utilisateur racine fédéré : un utilisateur racine se fédère à l'aide de l'opération `GetFederationToken` qui donne lieu à une session d'utilisateur fédéré pour cet utilisateur racine.

Lorsqu'un utilisateur IAM ou root demande des informations d'identification temporaires pour AWS STS utiliser cette opération, il lance une session utilisateur fédérée temporaire. L'ARN de cette séance est basé sur l'identité originale qui a été fédérée.

Pour spécifier l'ARN de la séance d'utilisateur fédéré dans l'élément `Principal`, utilisez le format suivant :

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:federated-user/user-name" }
```

AWS principes de service

Vous pouvez spécifier AWS des services dans l'`Principal`élément d'une politique basée sur les ressources ou dans des clés de condition qui prennent en charge les principes. Un principal de service est un identifiant pour un service.

Les rôles IAM qui peuvent être assumés par un AWS service sont appelés rôles ??? de service. Les rôles de service doivent inclure une politique d'approbation. Les politiques d'approbation sont des politiques basées sur les ressources attachées à un rôle qui définit les principaux qui peuvent endosser le rôle. Certains rôles de service disposent de politiques d'approbation prédéfinies. Toutefois, dans certains cas, vous devez spécifier le principal du service dans la politique d'approbation. Le principal de service d'une politique IAM ne peut pas être `"Service": "*"`.

> **Important**
>
> L'identifiant d'un principal de service inclut le nom du service et se présente généralement dans le format suivant :
> *service-name*`.amazonaws.com`

Le principal de service est défini par le service. Vous pouvez trouver le principal de service pour certains services en ouvrantAWS services qui fonctionnent avec IAM, en vérifiant si le service comporte Oui dans la colonne des Service-linked rôles, puis en ouvrant le lien Oui pour afficher la documentation des rôles liés au service pour ce service. Accédez à la section Autorisations de Service-Linked rôle pour ce service pour afficher le principal du service.

L'exemple suivant illustre une politique qui est peut être attachée à un rôle de service. La politique permet à deux services, Amazon ECS et Elastic Load Balancing, d'endosser le rôle. Les services peuvent ensuite effectuer toutes les tâches autorisées par la politique d'autorisation affectée au rôle (non illustré). Pour définir plusieurs principaux de service, vous ne spécifiez pas deux éléments `Service` ; vous ne devez en avoir qu'un seul. À la place, vous utilisez un tableau de plusieurs principaux de service comme valeur d'un élément `Service` unique.

```
"Principal": {
    "Service": [
        "ecs.amazonaws.com",
        "elasticloadbalancing.amazonaws.com"
   ]
}
```

AWS responsables de service dans les régions optionnelles

Vous pouvez lancer des ressources dans plusieurs AWS régions et vous devez vous inscrire à certaines de ces régions. Pour obtenir la liste complète des régions auxquelles vous devez vous inscrire, consultez la section Gestion des AWS régions dans le Références générales AWS guide.

Lorsqu'un AWS service d'une région optionnelle fait une demande dans la même région, le format du nom principal du service est identifié comme la version non régionalisée de son nom principal de service :

service-name`.amazonaws.com`

Lorsqu'un AWS service d'une région optionnelle envoie une demande interrégionale à une autre région, le format du nom principal du service est identifié comme étant la version régionalisée de son nom principal de service :

service-name`.`*{region}*`.amazonaws.com`

Par exemple, vous avez une rubrique Amazon SNS dans la région `ap-southeast-1` et un compartiment Amazon S3 dans la région d'adhésion `ap-east-1`. Vous voulez configurer les notifications du compartiment S3 pour publier des messages dans la rubrique SNS. Pour autoriser le service S3 à publier des messages sur la rubrique SNS, vous devez accorder au service S3 l'`sns:Publish`autorisation principale par le biais de la politique d'accès basée sur les ressources de la rubrique.

Si vous spécifiez la version non régionalisée du principal de service S3, `s3.amazonaws.com`, dans la stratégie d'accès à la rubrique, la demande `sns:Publish` du compartiment vers la rubrique

échouera. L'exemple suivant indique le principal de service S3 non régionalisé dans l'élément de stratégie `Principal` de la stratégie d'accès à la rubrique SNS.

```
"Principal": { "Service": "s3.amazonaws.com" }
```

Étant donné que le compartiment se trouve dans une région d'adhésion et que la demande est faite en dehors de cette même région, le principal de service S3 apparaît comme le nom du principal de service régionalisé, `s3.ap-east-1.amazonaws.com`. Vous devez utiliser le nom principal du service régionalisé lorsqu'un AWS service d'une région optionnelle adresse une demande à une autre région. Une fois que vous avez spécifié le nom du principal de service régionalisé, si le compartiment adresse une demande `sns:Publish` à la rubrique SNS située dans une autre région, la demande sera réussie. L'exemple suivant indique le principal de service S3 régionalisé dans l'élément de stratégie `Principal` de la stratégie d'accès à la rubrique SNS.

```
"Principal": { "Service": "s3.ap-east-1.amazonaws.com" }
```

Les stratégies de ressources ou les listes d'autorisation basées sur le principal de service pour les demandes inter-région d'une région d'adhésion vers une autre région ne seront réussies que si vous spécifiez le nom du principal de service régionalisé.

> **Note**
>
> Pour les stratégies d'approbation de rôle IAM, nous recommandons d'utiliser le nom du principal de service non régionalisé. Les ressources IAM sont globales et le même rôle peut donc être utilisé dans n'importe quelle région.

Tous les principaux

Vous pouvez utiliser un caractère générique (*) pour spécifier tous les principaux dans l'élément `Principal` d'une politique basée sur les ressources ou dans les clés de condition prenant en charge les principaux. Resource-based politiques accorde des autorisations et les clés de condition sont utilisées pour limiter les conditions d'une déclaration de politique.

> **Important**
>
> Nous vous recommandons fortement de ne pas utiliser de caractère générique (*) dans l'élément `Principal` d'une politique basée sur les ressources avec un effet `Allow`, sauf si

vous avez l'intention d'accorder un accès public ou anonyme. Sinon, indiquez les principaux, les services ou les comptes AWS visés dans l'élément `Principal`, puis restreignez davantage l'accès dans l'élément `Condition`. Cela est particulièrement vrai pour les politiques de confiance des rôles IAM, car elles permettent à d'autres principaux de devenir un principal dans votre compte.

Pour les politiques basées sur les ressources, utiliser un caractère générique (*) avec un effet `Allow` accorde l'accès à tous les utilisateurs, y compris les utilisateurs anonymes (accès public). Pour les utilisateurs IAM et les principaux de rôle de votre compte, aucune autre autorisation n'est requise. Pour les principaux d'autres comptes, ils doivent également disposer d'autorisations basées sur l'identité dans leur compte qui les autorise à accéder à votre ressource. C'est ce que l'on appelle l'accès entre comptes.

Pour les utilisateurs anonymes, les éléments suivants sont équivalents :

```
"Principal": "*"
```

```
"Principal" : { "AWS" : "*" }
```

Vous ne pouvez pas utiliser un caractère générique pour une correspondance à une partie d'un nom ou d'un ARN principal.

L'exemple suivant montre une politique basée sur les ressources qui peut être utilisée à la place de AWS Éléments de politique JSON : NotPrincipal dans le but de refuser explicitement tous les principaux, à l'exception de ceux qui sont spécifiés dans l'élément `Condition`. Cette politique doit être ajoutée à un compartiment Amazon S3.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "UsePrincipalArnInsteadOfNotPrincipalWithDeny",
      "Effect": "Deny",
      "Action": "s3:*",
      "Principal": "*",
```

```
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/*",
        "arn:aws:s3:::amzn-s3-demo-bucket"
      ],
      "Condition": {
        "ArnNotEquals": {
          "aws:PrincipalArn": "arn:aws:iam::444455556666:user/user-name"
        }
      }
    }
  ]
}
```

En savoir plus

Pour plus d'informations, consultez les ressources suivantes :

- Exemples de politique de compartiment dans le Guide de l'utilisateur service de stockage simple Amazon
- Exemples de politiques pour Amazon SNS dans le Guide du développeur Amazon Simple Notification Service
- Exemples de politique Amazon SQS dans le Guide du développeur Amazon Simple Queue Service
- Politiques de clé dans le Guide du développeur AWS Key Management Service
- Identifiants de compte (français non garanti) dans Références générales AWS
- Fédération OIDC

AWS Éléments de politique JSON : NotPrincipal

L'élément `NotPrincipal` utilise `"Effect":"Deny"` pour refuser l'accès à tous les principaux, sauf le principal spécifié dans l'élément `NotPrincipal`. Un principal peut être un utilisateur IAM, un utilisateur principal AWS STS fédéré, un rôle IAM, une session de rôle assumé Compte AWS, un AWS service ou un autre type principal. Pour plus d'informations sur les principaux, consultez AWS Éléments de politique JSON : Principal.

L'élément `NotPrincipal` doit être utilisé avec `"Effect":"Deny"`. Son utilisation avec `"Effect":"Allow"` n'est pas prise en charge.

> **Important**
>
> Nous ne recommandons pas l'utilisation de `NotPrincipal`pour les nouvelles politiques basées sur les ressources dans le cadre de votre politique de sécurité et d'autorisation. Lorsque vous utilisez l'élément `NotPrincipal`, il peut être difficile de résoudre les problèmes liés à plusieurs types de politique. Nous recommandons plutôt l'utilisation de la clé de contexte `aws:PrincipalArn` avec les opérateurs de condition ARN.

Points clés

- L'élément `NotPrincipal` est pris en charge dans des politiques basées sur les ressources pour certains services AWS , y compris les points de terminaison d'un VPC. Les politiques basées sur les ressources sont des politiques que vous intégrez directement à une ressource. Vous ne pouvez pas utiliser l'élément `NotPrincipal` dans une politique IAM basée sur l'identité ni dans une politique d'approbation du rôle IAM.
- N'utilisez pas de déclarations de politique basée sur les ressources qui incluent un élément de politique `NotPrincipal` ayant un effet `Deny` sur les utilisateurs IAM ou les rôles auxquels est attachée une politique de limite des autorisations. L'élément `NotPrincipal` ayant un effet `Deny` refusera toujours tout principal IAM auquel est attachée une politique de limite des autorisations, quelles que soient les valeurs spécifiées dans l'élément `NotPrincipal`. Certains utilisateurs ou rôles IAM qui auraient autrement eu accès à la ressource n'y ont donc plus accès. Nous vous recommandons de modifier vos déclarations de politique basée sur les ressources afin d'utiliser l'opérateur de condition `ArnNotEquals` avec la clé de contexte `aws:PrincipalArn` dans le but de limiter l'accès plutôt que l'élément `NotPrincipal`. Pour plus d'informations sur les limites des autorisations, veuillez consulter Limites d'autorisations pour les entités IAM.
- Lorsque vous utilisez `NotPrincipal`, vous devez également spécifier l'ARN du compte du principal non refusé. Dans le cas contraire, la politique peut refuser l'accès à l'ensemble du compte contenant le principal. En fonction du service que vous incluez dans votre politique, AWS peut valider d'abord le compte, puis l'utilisateur. Si un utilisateur assumant un rôle (une personne utilisant un rôle) est en cours d'évaluation, AWS vous pouvez d'abord valider le compte, puis le rôle, puis l'utilisateur assumé. Ce dernier est identifié par le nom de session de rôle qui est spécifié lorsqu'il endosse le rôle. Par conséquent, nous vous recommandons vivement d'inclure explicitement l'ARN du compte d'un utilisateur, ou inclure à la fois l'ARN d'un rôle et l'ARN du compte contenant le rôle.

- L'élément `NotPrincipal` n'est pas pris en charge dans les politiques de contrôle des services (SCP) et les politiques de contrôle des ressources (RCP).

Alternatives à l'élément **`NotPrincipal`**

Lorsque vous gérez le contrôle d'accès dans AWS, il peut arriver que vous deviez refuser explicitement à tous les principaux l'accès à une ressource, à l'exception d'un ou de plusieurs principaux que vous spécifiez. AWS recommande d'utiliser une instruction Deny avec des clés contextuelles de condition globales pour un contrôle plus précis et un dépannage simplifié. Les exemples suivants montrent d'autres approches utilisant des opérateurs de condition tels que `StringNotEquals` ou `ArnNotEquals` pour refuser l'accès à tous les principaux, à l'exception de ceux spécifiés dans l'élément Condition.

Exemple de scénario utilisant un rôle IAM

Vous pouvez utiliser une politique basée sur les ressources avec une instruction Refuser pour empêcher tous les rôles IAM, à l'exception de ceux spécifiés dans l'élément Condition, d'accéder à vos ressources ou de les manipuler. Cette approche suit le principe AWS de sécurité selon lequel un refus explicite a toujours priorité sur toute instruction d'autorisation et contribue à maintenir le principe du moindre privilège dans l'ensemble de votre AWS infrastructure.

Au lieu d'utiliser `NotPrincipal`, nous vous recommandons d'utiliser une instruction Refuser avec des clés de contexte de condition globales et l'opérateur de condition comme `ArnNotEquals` pour autoriser explicitement un rôle IAM à accéder à vos ressources. L'exemple suivant utilise aws : PrincipalArn pour autoriser explicitement le rôle `read-only-role` à accéder aux compartiments Amazon S3 dans le dossier `Bucket_Account_Audit`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyCrossAuditAccess",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::Bucket_Account_Audit",
```

```
        "arn:aws:s3:::Bucket_Account_Audit/*"
      ],
      "Condition": {
        "ArnNotEquals": {
          "aws:PrincipalArn": "arn:aws:iam::444455556666:role/read-only-role"
        }
      }
    }
  ]
}
```

Exemple de scénario utilisant un principal de service

Vous pouvez utiliser une instruction Refuser pour empêcher tous les principaux de service, à l'exception de ceux spécifiés dans l'élément `Condition`, d'accéder à vos ressources ou de les manipuler. Cette approche est particulièrement utile lorsque vous devez mettre en œuvre des contrôles d'accès précis ou établir des limites de sécurité entre différents services et applications dans votre environnement AWS .

Au lieu d'utiliser `NotPrincipal`, nous vous recommandons d'utiliser une instruction Refuser avec des clés de contexte de condition globales et l'opérateur de condition `StringNotEquals` pour autoriser explicitement un principal de service à accéder à vos ressources. L'exemple suivant utilise `aws : PrincipalServiceName` pour autoriser explicitement le principal de service AWS CodeBuild à accéder aux compartiments Amazon S3 dans le dossier `BUCKETNAME`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyNotCodeBuildAccess",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::BUCKETNAME",
        "arn:aws:s3:::BUCKETNAME/*"
      ],
      "Condition": {
```

```
        "StringNotEqualsIfExists": {
          "aws:PrincipalServiceName": "codebuild.amazonaws.com"
        }
      }
    }
  ]
}
```

Éléments de la politique JSON IAM : Action

L'élément `Action` décrit les actions spécifiques qui seront accordées ou refusées. Les instructions doivent inclure un élément `Action` ou `NotAction`. Chaque AWS service possède son propre ensemble d'actions qui décrivent les tâches que vous pouvez effectuer avec ce service. Par exemple, la liste des actions pour Amazon S3 se trouve dans la section Spécifier les autorisations dans une politique du guide de l'utilisateur d'Amazon Simple Storage Service, la liste des actions pour Amazon EC2 se trouve dans la référence d'API Amazon EC2 et la liste des actions pour se Gestion des identités et des accès AWS trouve dans la référence d'API IAM. Pour consulter la liste d'actions d'autres services, reportez-vous à la documentation de référence de l'API pour le service requis.

AWS fournit également des informations de référence de service au format JSON afin de rationaliser l'automatisation des flux de travail de gestion des politiques. Grâce aux informations de référence du service, vous pouvez accéder aux actions, ressources et clés de condition disponibles Services AWS à partir de fichiers lisibles par machine. Pour plus d'informations, consultez les Informations sur les Service AWS simplifiés pour l'accès programmatique dans la Référence des autorisations de service.

Vous spécifiez une valeur en utilisant un espace de noms du service comme préfixe d'action (`iam`, `ec2` `sqs`, `sns`, `s3`, etc.), suivi du nom de l'action à autoriser ou refuser. Le nom doit correspondre à une action prise en charge par le service. Le préfixe et le nom d'action ne sont pas sensibles à la casse. Par exemple, `iam:ListAccessKeys` est identique à `IAM:listaccesskeys`. Les exemples suivants illustrent les éléments `Action` pour différents services.

Action Amazon SQS

```
"Action": "sqs:SendMessage"
```

Action Amazon EC2

```
"Action": "ec2:StartInstances"
```

Action IAM

```
"Action": "iam:ChangePassword"
```

Actions Amazon S3

```
"Action": "s3:GetObject"
```

Vous pouvez spécifier plusieurs valeurs pour l'élément `Action`.

```
"Action": [ "sqs:SendMessage", "sqs:ReceiveMessage", "ec2:StartInstances",
 "iam:ChangePassword", "s3:GetObject" ]
```

Vous pouvez utiliser des caractères génériques de correspondance à plusieurs caractères (*) et des caractères génériques à caractère unique (?) pour donner accès à toutes les actions proposées par un produit spécifique. AWS Par exemple, l'élément `Action` suivant s'applique à toutes les actions S3.

```
"Action": "s3:*"
```

Vous pouvez aussi utiliser des caractères génériques (* ou ?) dans le nom d'action. Par exemple, l'élément `Action` suivant s'applique à toutes les actions IAM qui incluent la chaîne `AccessKey`, y compris `CreateAccessKey`, `DeleteAccessKey`, `ListAccessKeys` et `UpdateAccessKey`.

```
"Action": "iam:*AccessKey*"
```

Certains services vous permettent de limiter les actions disponibles. Par exemple, Amazon SQS vous permet de ne mettre à disposition qu'un sous-ensemble de toutes les actions Amazon SQS possibles. Dans ce cas, le caractère générique * ne permet pas de contrôler entièrement la file d'attente ; il autorise uniquement le sous-ensemble d'actions que vous avez partagées. Pour de plus amples informations, veuillez consulter Comprendre les autorisations dans le Guide du développeur Amazon Simple Storage Service.

Éléments de politique JSON IAM : NotAction

`NotAction` est un élément de politique avancé qui correspond de manière explicite à tout sauf à la liste spécifiée des actions. L'utilisation de `NotAction` peut entraîner une politique plus courte en

répertoriant uniquement quelques actions qui ne devraient pas correspondre, plutôt que d'inclure une longue liste d'actions qui correspondront. Les actions spécifiées dans `NotAction` ne sont pas affectées par l'effet `Allow` ou `Deny` d'une déclaration de politique. En revanche, cela signifie que toutes les actions ou services applicables non répertoriés sont autorisés si vous utilisez l'effet `Allow`. En outre, ces actions ou services non répertoriés sont refusées si vous utilisez l'effet `Deny`. Lorsque vous utilisez `NotAction` avec l'élément `Resource`, vous fournissez la portée de la politique. C'est ainsi que l'on AWS détermine les actions ou les services applicables. Pour plus d'informations, consultez l'exemple de politique suivant :

NotAction avec Allow

Vous pouvez utiliser l'`NotAction`élément dans une instruction `"Effect": "Allow"` pour donner accès à toutes les actions d'un AWS service, à l'exception des actions spécifiées dans`NotAction`. Vous pouvez l'utiliser avec l'élément `Resource` pour fournir la portée de la politique, en limitant les actions autorisées aux actions pouvant être exécutées sur la ressource spécifiée.

L'exemple suivant permet aux utilisateurs d'accéder à toutes les actions Amazon S3 pouvant être exécutées sur une ressource S3 sauf pour supprimer un compartiment. Cette politique n'autorise pas non plus les actions dans d'autres services, car les actions d'autres services ne sont pas applicables aux ressources S3.

```
"Effect": "Allow",
"NotAction": "s3:DeleteBucket",
"Resource": "arn:aws:s3:::*",
```

Il peut arriver que vous souhaitiez autoriser l'accès à un grand nombre d'actions. En utilisant l'élément `NotAction`, vous inversez de manière efficace l'instruction, ce qui réduit la liste des actions. Par exemple, en raison AWS du grand nombre de services, vous souhaiterez peut-être créer une politique permettant à l'utilisateur de tout faire sauf d'accéder aux actions IAM.

L'exemple suivant permet aux utilisateurs d'accéder à toutes les actions de tous les AWS services, à l'exception d'IAM.

```
"Effect": "Allow",
"NotAction": "iam:*",
"Resource": "*"
```

Soyez vigilant lorsque vous utilisez l'élément `NotAction` et `"Effect": "Allow"` dans la même instruction ou une instruction différente dans une politique. `NotAction` correspond à tous les

services et les actions qui ne sont pas répertoriés de manière explicite ou applicables à la ressource spécifiée, et peut accorder aux utilisateurs plus d'autorisations que vous n'auriez souhaité.

NotAction avec Deny

Vous pouvez utiliser l'élément `NotAction` dans une instruction avec `"Effect": "Deny"` pour refuser un accès à l'ensemble des ressources répertoriées sauf pour les actions spécifiées dans l'élément `NotAction`. Cette combinaison n'autorise pas les éléments répertoriés, mais à la place elle refuse de manière explicite les actions non répertoriées. Vous devez toujours autoriser les actions que vous souhaitez autoriser.

L'exemple conditionnel suivant refuse l'accès aux actions non-IAM si l'utilisateur n'est pas connecté lorsqu'il utilise MFA. Si l'utilisateur est connecté lorsqu'il utilise MFA, le test `"Condition"` échoue et l'instruction `"Deny"` finale n'a aucun effet. Notez, toutefois, que ceci n'accorde à l'utilisateur l'accès à aucune action, mais ne fait que refuser explicitement toutes les autres actions sauf les actions IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Sid": "DenyAllUsersNotUsingMFA",
        "Effect": "Deny",
        "NotAction": "iam:*",
        "Resource": "*",
        "Condition": {"BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}}
    }]
}
```

Pour obtenir un exemple de politique qui refuse l'accès aux actions en dehors de régions spécifiques, à l'exception des actions provenant de services spécifiques, veuillez consulter AWS: refuse l'accès AWS en fonction de la région demandée.

Éléments de la politique JSON IAM : Ressource

L'élément `Resource` d'une déclaration de politique IAM définit l'objet ou les objets auxquels la déclaration s'applique. Les instructions doivent inclure un élément `Resource` ou `NotResource`.

Vous indiquez une ressource à l'aide d'un Amazon Resources Name (ARN). Le format de l'ARN dépend de Service AWS la ressource spécifique à laquelle vous faites référence. Bien que le format

de l'ARN varie, vous utilisez toujours un ARN pour identifier une ressource. Pour de plus amples informations sur le format des ARN, veuillez consulter ARN IAM. Pour plus d'informations sur la façon de spécifier une ressource, reportez-vous à la documentation du service pour lequel vous souhaitez rédiger une instruction.

> **Note**
>
> Certains Services AWS ne vous permettent pas de spécifier des actions pour des ressources individuelles. Dans ces cas, toutes les actions répertoriées dans `Action`ou l'élément `NotAction` s'appliquent à toutes les ressources de ce service. Dans ce cas, vous utilisez le caractère générique (*) dans l'élément `Resource`.

L'exemple suivant fait référence à une file d'attente Amazon SQS spécifique.

```
"Resource": "arn:aws:sqs:us-east-2:account-ID-without-hyphens:queue1"
```

L'exemple suivant fait référence à un utilisateur IAM nommé Bob dans un Compte AWS.

> **Note**
>
> Dans l'élément `Resource`, le nom de l'utilisateur IAM est sensible à la casse.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/Bob"
```

Utilisation de caractères génériques dans les ARN de ressources

Vous pouvez utiliser des caractères génériques (* et ?) dans les segments individuels d'un ARN (les parties séparées par des deux points) pour représenter :

- Toute combinaison de caractères (*)
- Tout caractère unique (?)

Vous pouvez utiliser plusieurs * ou ? dans chaque segment. Si le caractère générique * est le dernier caractère d'un segment d'ARN de ressource, il peut s'étendre pour correspondre au-delà des limites des deux points. Nous vous recommandons d'utiliser des caractères génériques (* et ?) dans les segments d'ARN séparés par deux points.

> **Note**
>
> Vous ne pouvez pas utiliser de caractère générique dans le segment de service qui identifie le AWS produit. Pour plus d'informations sur les segments ARN, consultez Identifier AWS ressources avec Amazon Resource Names (ARN)

L'exemple suivant fait référence à tous les utilisateurs IAM dont le chemin d'accès est `/accounting`.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/accounting/*"
```

L'exemple suivant fait référence à tous les éléments d'un compartiment Amazon S3 spécifique.

```
"Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
```

Le caractère astérisque (*) peut se développer pour remplacer tout ce qui se trouve dans un segment, y compris des caractères tels qu'une barre oblique (/) qui pourrait autrement sembler être un délimiteur dans un espace de noms de service donné. Par exemple, considérez l'ARN Amazon S3 suivant comme la même logique d'extension générique qui s'applique à tous les services.

```
"Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*/test/*"
```

Les caractères génériques de l'ARN s'appliquent à tous les objets suivants dans le compartiment, et pas seulement au premier objet répertorié.

```
amzn-s3-demo-bucket/1/test/object.jpg
amzn-s3-demo-bucket/1/2/test/object.jpg
amzn-s3-demo-bucket/1/2/test/3/object.jpg
amzn-s3-demo-bucket/1/2/3/test/4/object.jpg
amzn-s3-demo-bucket/1///test///object.jpg
amzn-s3-demo-bucket/1/test/.jpg
amzn-s3-demo-bucket//test/object.jpg
amzn-s3-demo-bucket/1/test/
```

Considérez les deux derniers objets de la liste précédente. Un nom d'objet Amazon S3 peut commencer ou se terminer par la barre oblique (/) du délimiteur classique. Bien que / fonctionne comme un délimiteur, ce caractère n'a pas de signification spécifique lorsqu'il est utilisé dans un ARN de ressource. Il est traité de la même manière que n'importe quel autre caractère valide. L'ARN ne correspondrait pas aux objets suivants :

```
amzn-s3-demo-bucket/1-test/object.jpg
amzn-s3-demo-bucket/test/object.jpg
amzn-s3-demo-bucket/1/2/test.jpg
```

Spécification de plusieurs ressources

Vous pouvez spécifier plusieurs ressources dans l'élément `Resource` à l'aide d'un tableau d'ARN. L'exemple suivant fait référence à deux tables DynamoDB.

```
"Resource": [
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/books_table",
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/magazines_table"
]
```

Utilisation de variables de politique dans les ARN de ressources

Dans l'élément `Resource`, vous pouvez utiliser des variables de politique JSON dans la partie de l'ARN qui identifie la ressource spécifique (autrement dit, dans la partie finale de l'ARN). Par exemple, vous pouvez utilisez la clé `{aws:username}` au sein de l'ARN de la ressource pour indiquer que le nom de l'utilisateur actuel doit être inclus dans le nom de la ressource. L'exemple suivant montre comment utiliser la clé `{aws:username}` dans un élément `Resource`. La politique autorise l'accès à une table Amazon DynamoDB correspondant au nom de l'utilisateur actuel.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "dynamodb:*",
        "Resource": "arn:aws:dynamodb:us-east-2:111122223333:table/
${aws:username}"
    }
}
```

Pour plus d'informations sur les variables de politique JSON, consultez Éléments des politiques IAM : variables et balises.

Éléments de la politique JSON IAM : NotResource

`NotResource` est un élément de politique avancé qui correspond explicitement à chaque ressource, à l'exception de celles spécifiées. L'utilisation de `NotResource` peut entraîner une politique plus courte en répertoriant uniquement quelques ressources qui ne devraient pas correspondre, plutôt que d'inclure une longue liste de ressources qui correspondront. Ceci est particulièrement utile pour les politiques applicables au sein d'un même service AWS .

Prenons l'exemple d'un groupe nommé `HRPayroll`. Les membres de `HRPayroll` ne doivent pas être autorisés à accéder aux ressources Amazon S3 sauf au dossier `Payroll` dans le compartiment `HRBucket`. La politique suivante refuse de manière explicite l'accès à toutes les ressources Amazon S3 autres que celles répertoriées. Notez, toutefois, que cette politique n'accorde à l'utilisateur aucun accès aux ressources.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "s3:*",
    "NotResource": [
      "arn:aws:s3:::HRBucket/Payroll",
      "arn:aws:s3:::HRBucket/Payroll/*"
    ]
  }
}
```

En règle générale, pour refuser de manière explicite l'accès à une ressource vous rédigez une politique qui utilise `"Effect":"Deny"` et inclut un élément `Resource` qui répertorie individuellement chaque dossier. Toutefois, dans ce cas, chaque fois que vous ajoutez un dossier à `HRBucket`, ou une ressource à Amazon S3 à laquelle personne ne doit accéder, vous devez ajouter son nom à la liste dans l'élément `Resource`. Si vous utilisez un élément `NotResource` à la place, les utilisateurs verront leur accès aux nouveaux dossiers automatiquement refusé sauf si vous ajoutez les noms des dossiers à l'élément `NotResource`.

Lorsque vous utilisez `NotResource`, n'oubliez pas que les ressources spécifiées dans cet élément sont les seules ressources qui ne sont pas limitées. Toutes les ressources qui s'appliqueraient à

l'action sont limitées. Dans l'exemple ci-dessus, la politique affecte uniquement les actions Amazon S3, et donc uniquement les ressources Amazon S3. Si l'élément `Action` comprenait également des actions Amazon EC2, la politique refuserait alors l'accès à toutes les ressources EC2 non spécifiées dans l'élément `NotResource`. Pour savoir quelles actions d'un service permettent de spécifier l'ARN d'une ressource, consultez la section Actions, ressources et clés de condition pour les AWS services.

NotResource avec d'autres éléments

Vous ne devez jamais utiliser les éléments `"Effect": "Allow"`, `"Action": "*"` et `"NotResource": "arn:aws:s3:::HRBucket"` ensemble. Cette instruction est très dangereuse, car elle autorise toutes les actions AWS sur toutes les ressources, à l'exception du compartiment `HRBucket` S3. Elle autoriserait même l'utilisateur à s'ajouter une politique qui lui permettrait d'accéder à `HRBucket`. Ne le faites pas.

Soyez vigilant lorsque vous utilisez l'élément `NotResource` et `"Effect": "Allow"` dans la même instruction ou une instruction différente dans une politique. `NotResource` autorise tous les services et ressources qui ne sont pas répertoriés de manière explicite et peut accorder aux utilisateurs plus d'autorisations que vous n'auriez souhaité. L'utilisation de l'élément `NotResource` et `"Effect": "Deny"` dans la même instruction refuse les services et les ressources qui ne sont pas répertoriés de manière explicite.

Éléments de la politique JSON IAM : Condition

L'élément `Condition` (ou bloc `Condition`) vous permet de spécifier des conditions lorsqu'une politique est appliquée. L'élément `Condition` est facultatif. Dans l'élément `Condition`, vous créez des expressions dans lesquelles vous utilisez des opérateurs de condition (égal, inférieur à, etc.) pour faire correspondre les clés et valeurs de contexte de la politique avec les clés et valeurs du contexte de la demande. Pour de plus amples informations sur le contexte de la demande, veuillez consulter Composants d'une requête.

```
"Condition" : { "{condition-operator}" : { "{condition-key}" : "{condition-value}" }}
```

La clé de contexte que vous spécifiez dans une condition de politique peut être une clé de contexte de condition globale ou une clé de contexte spécifique au service. Les clés de contexte des conditions globales ont le `aws:` préfixe. Service-specific les clés de contexte ont le préfixe du service. Par exemple, Amazon EC2 vous permet d'écrire une condition à l'aide de la clé de contexte `ec2:InstanceType`, qui est propre à ce service. Pour connaître les clés de contexte IAM

spécifiques au service ayant le préfixe `iam:`, veuillez consulter la rubrique IAM et AWS STS clés de contexte de condition.

Les noms de clé de contexte ne sont pas sensibles à la casse. Par exemple, l'inclusion de la clé de `aws:SourceIP` contexte revient à tester pour`AWS:SourceIp`. Case-sensitivity des valeurs de clé contextuelle dépend de l'opérateur de condition que vous utilisez. Par exemple, la condition suivante inclut l'opérateur `StringEquals` pour garantir que seules les demandes effectuées par `john` correspondent. Les utilisateurs nommés `John` se voient refuser l'accès.

```
"Condition" : { "StringEquals" : { "aws:username" : "john" }}
```

La condition suivante utilise l'opérateur `StringEqualsIgnoreCase` pour la correspondance avec les utilisateurs nommés `john` ou `John`.

```
"Condition" : { "StringEqualsIgnoreCase" : { "aws:username" : "john" }}
```

Certaines clés de contexte prennent en charge des paires clé-valeur qui vous permettent de spécifier une partie du nom de clé. Les exemples incluent la clé de `aws:RequestTag/tag-key` contexte AWS KMS `kms:EncryptionContext:encryption_context_key`, la clé de contexte et la clé de `ResourceTag/tag-key` contexte prise en charge par plusieurs services.

- Si vous utilisez la clé de contexte `ResourceTag/tag-key` pour un service comme Amazon EC2, vous devez spécifier un nom de clé pour `tag-key`.
- Les noms de clé ne sont pas sensibles à la casse. Cela signifie que si vous spécifiez `"aws:ResourceTag/TagKey1": "Value1"` dans l'élément de condition de votre politique, la condition correspond à une clé de balise de ressource nommée `TagKey1` ou `tagkey1`, mais pas aux deux.
- AWS les services qui prennent en charge ces attributs peuvent vous permettre de créer plusieurs noms de clé qui ne diffèrent que majuscules et minuscules. Par exemple, vous pouvez baliser une instance Amazon EC2 avec les interfaces `ec2=test1` et `EC2=test2`. Lorsque vous utilisez une condition comme `"aws:ResourceTag/EC2": "test1"` pour autoriser l'accès à cette ressource, le nom de clé correspond aux deux balises, mais une seule valeur correspond. Cela peut entraîner des échecs de condition inattendus.

> **Important**
>
> En tant que bonne pratique, assurez-vous que les membres de votre compte suivent une convention de dénomination cohérente pour les attributs avec paire clé-valeur. Les exemples incluent les balises ou les contextes de chiffrement AWS KMS . Vous pouvez l'appliquer à l'aide de la clé de `aws:TagKeys` contexte pour le balisage ou de la clé `kms:EncryptionContextKeys` pour le contexte de AWS KMS chiffrement.

- Pour obtenir une liste de tous les opérateurs de condition et une description de leur fonctionnement, veuillez consulter la rubrique Opérateurs de condition.
- Sauf indication contraire, toutes les clés de contexte peuvent comporter plusieurs valeurs. Pour savoir comment traiter des clés de contexte qui ont plusieurs valeurs, veuillez consulter la rubrique Définir les opérateurs pour les clés de contexte à valeurs multiples.
- Pour obtenir la liste de l'ensemble des clés de contexte disponibles dans le monde entier, veuillez consulter la rubrique AWS clés de contexte de condition globale.
- Pour les clés de contexte conditionnelles définies par chaque service, voir Actions, ressources et clés de condition pour les AWS services.

Contexte de la demande

Lorsqu'un responsable fait une demande à AWS, AWS rassemble les informations de demande dans un contexte de demande. Le contexte de la demande comprend des informations sur le principal, les ressources, les actions et d'autres propriétés environnementales. L'évaluation de la politique fait correspondre les propriétés de la politique aux propriétés envoyées dans la demande afin d'évaluer et d'autoriser les actions que vous pouvez effectuer dans AWS.

Vous pouvez utiliser l'élément `Condition` d'une politique JSON pour tester des clés de contexte spécifiques par rapport au contexte de la demande. Par exemple, vous pouvez créer une politique qui utilise la clé de CurrentTime contexte aws : pour permettre à un utilisateur d'effectuer des actions uniquement dans un intervalle de dates spécifique.

L'exemple suivant montre une représentation du contexte de la demande lorsque Martha Rivera envoie une demande pour désactiver son dispositif MFA.

```
Principal: AROA123456789EXAMPLE
Action: iam:DeactivateMFADevice
```

```
Resource: arn:aws:iam::user/martha
Context:
  - aws:UserId=AROA123456789EXAMPLE:martha
  - aws:PrincipalAccount=1123456789012
  - aws:PrincipalOrgId=o-example
  - aws:PrincipalARN=arn:aws:iam::1123456789012:assumed-role/TestAR
  - aws:MultiFactorAuthPresent=true
  - aws:MultiFactorAuthAge=2800
  - aws:CurrentTime=...
  - aws:EpochTime=...
  - aws:SourceIp=...
```

Le contexte de la demande est mis en correspondance avec une politique qui autorise les utilisateurs à supprimer leur propre dispositif authentification multifactorielle (MFA), mais uniquement s'ils se sont connectés à l'aide de la MFA au cours de la dernière heure (3 600 secondes).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Sid": "AllowRemoveMfaOnlyIfRecentMfa",
        "Effect": "Allow",
        "Action": [
            "iam:DeactivateMFADevice"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}",
        "Condition": {
            "NumericLessThanEquals": {"aws:MultiFactorAuthAge": "3600"}
        }
    }
}
```

Dans cet exemple, la politique correspond au contexte de la demande : l'action est la même, la ressource correspond au caractère générique « * » et la valeur pour `aws:MultiFactorAuthAge` est 2 800, qui est inférieure à 3 600. La politique autorise donc cette demande d'autorisation.

AWS évalue chaque clé de contexte de la politique et renvoie une valeur vraie ou fausse. L'absence de clé de contexte dans la demande est considérée comme une absence de correspondance.

Le contexte de la demande peut renvoyer les valeurs suivantes :

- True : si le demandeur s'est connecté avec MFA au cours de la dernière heure ou moins, la condition renvoie la valeur true.
- False : si le demandeur s'est connecté avec MFA il y a plus d'une heure, la condition renvoie la valeur false.
  - Absente : si le demandeur a fait une demande à l'aide de ses clés d'accès utilisateur IAM dans l' AWS API AWS CLI ou, la clé n'est pas présente. Dans ce cas, la clé est manquante et il n'y a pas de correspondance.

> **Note**
>
> Dans certains cas, lorsque la valeur de la clé de condition n'est pas présente, la condition peut toujours retourner vraie. Par exemple, si vous ajoutez le qualificateur `ForAllValues`, la demande renvoie vrai si la clé de contexte ne figure pas dans la demande. Pour éviter que des clés de contexte manquantes ou des clés de contexte contenant des valeurs vides ne soient considérées comme vraies, vous pouvez inclure l'opérateur de condition Null dans votre politique avec une valeur `false` pour vérifier si la clé de contexte existe et si sa valeur n'est pas nulle.

Bloc Condition

L'exemple suivant illustre le format de base d'un élément `Condition` :

```
"Condition": {"StringLike": {"s3:prefix": ["jane/*"]}}
```

Une valeur de la demande est représentée par une clé de contexte. Dans ce cas, il s'agit de `s3:prefix`. La valeur clé de contexte est comparée à une valeur que vous spécifiez comme valeur littérale, par exemple `jane/*`. Le type de comparaison à effectuer est spécifié par l'opérateur de condition (ici, `StringLike`). Vous pouvez créer des conditions qui comparent des chaînes, des dates, des numéros et autres à l'aide d'opérateurs de comparaison booléens standard comme est égal à, supérieur à ou inférieur à. Lorsque vous utilisez des opérateurs de chaîne ou des opérateurs ARN, vous pouvez également utiliser une variable de politique dans la valeur de clé de contexte. L'exemple suivant inclut la variable `aws:username`.

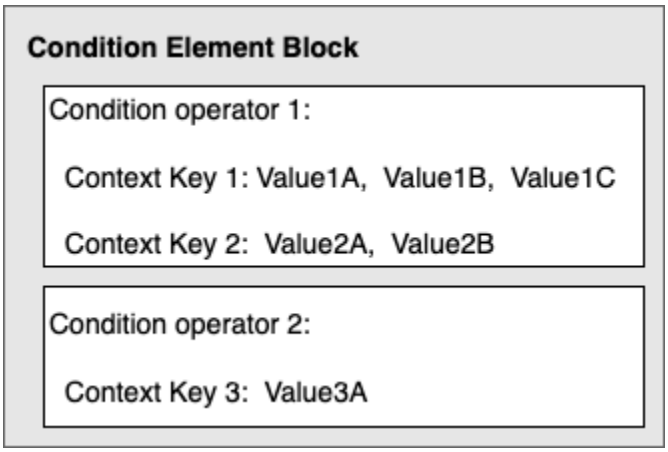
```
"Condition": {"StringLike": {"s3:prefix": ["${aws:username}/*"]}}
```

Dans certains cas, les clés de contexte peuvent contenir plusieurs valeurs. Par exemple, une demande à Amazon DynamoDB peut retourner ou mettre à jour plusieurs attributs d'une table. Une politique d'accès aux tables DynamoDB peut inclure la clé de contexte `dynamodb:Attributes`, qui contient tous les attributs spécifiés dans la demande. Vous pouvez tester les divers attributs de la demande par rapport à une liste d'attributs autorisés dans une politique à l'aide d'opérateurs de définition dans l'élément `Condition`. Pour de plus amples informations, veuillez consulter Définir les opérateurs pour les clés de contexte à valeurs multiples.

Lorsque la politique est évaluée lors d'une demande, AWS remplace la clé par la valeur correspondante de la demande. (Dans cet exemple, AWS utiliserait la date et l'heure de la demande.) L'évaluation de la condition retourne True ou False, ce qui est pris en compte pour déterminer si la politique dans sa totalité autorise ou refuse la demande.

Plusieurs valeurs dans un élément Condition

Un élément `Condition` peut contenir plusieurs opérateurs de condition, et chaque opérateur de condition peut également inclure plusieurs paires clé-valeur de contexte. L'illustration suivante décrit ce scénario.



Pour de plus amples informations, veuillez consulter Définir les opérateurs pour les clés de contexte à valeurs multiples.

Éléments de politique JSON IAM : Opérateurs de condition

Utilisez les opérateurs de condition de l'élément `Condition` pour faire correspondre la clé de condition et la valeur de la politique avec les valeurs du contexte de la demande. Pour en savoir plus sur l'élément `Condition`, consultez Éléments de la politique JSON IAM : Condition.

L'opérateur de condition que vous pouvez utiliser dans une politique dépend de la clé de condition que vous choisissez. Vous pouvez choisir une clé de condition globale ou une clé de condition spécifique au service. Pour savoir quel opérateur de condition vous pouvez utiliser pour une clé de condition globale, veuillez consulter AWS clés de contexte de condition globale. Pour savoir quel opérateur de condition vous pouvez utiliser pour une clé de condition spécifique à un service, voir Actions, ressources et clés de condition pour les AWS services et choisissez le service que vous souhaitez afficher.

> **Important**
>
> Si la clé que vous spécifiez dans une condition de stratégie n'est pas présente dans le contexte de la requête, les valeurs ne correspondent pas et la condition est fausse. Si la condition de stratégie requiert qu'il n'y ait aucune correspondance de clé, tels que `StringNotLike` ou `ArnNotLike` et la touche de droite n'est pas présente, la condition est vraie. Cette logique s'applique à tous les opérateurs de condition sauf... IfExistset Null check. Ces opérateurs testent si la clé est présente (existe) dans le contexte de demande.

Les opérateurs de condition peuvent être regroupés dans les catégories suivantes :

- String
- Numérique
- Date et heure
- Booléen
- Binaire
- Adresse IP
- Amazon Resource Name (ARN) (disponible uniquement pour certains services.)
- ... IfExists(vérifie si la valeur de clé existe dans le cadre d'une autre vérification)
- Null check (vérifie si la valeur de clé existe en tant que vérification autonome)

Opérateurs de condition de chaîne

Les opérateurs de condition de chaîne permettent de créer des éléments `Condition` qui limitent l'accès après comparaison d'une clé à une valeur de chaîne.

- Variables de politiques – Prises en charge

- Caractères génériques – Pris en charge

Opérateur de condition	Description
`StringEquals`	Correspondance exacte, respect de la casse
`StringNotEquals`	Correspondance négative
`StringEqualsIgnoreCase`	Correspondance exacte, non respect de la casse
`StringNotEqualsIgnoreCase`	Correspondance négative, non respect de la casse
`StringLike`	Case-sensitive correspondant. Les valeurs peuvent inclure un caractère générique (*) correspondant à plusieurs caractères et un caractère générique (?) correspondant à un seul caractère n'importe où dans la chaîne. Vous devez spécifier des caractères génériques pour obtenir des correspondances de chaînes partielles. **Note** Si une clé contient plusieurs valeurs, `StringLike` peut être qualifié avec les opérateurs d'ensemble `ForAllValues:StringLike` et `ForAnyValue:StringLike`. Pour de plus amples informations, veuillez consulter Définir les opérateurs pour les clés de contexte à valeurs multiples.
`StringNotLike`	Correspondance avec non respect de la casse. Les valeurs peuvent inclure un caractère générique (*) correspondant à plusieurs caractères ou un caractère générique (?) correspondant à un seul caractère n'importe où dans la chaîne.

Example opérateur de condition de chaîne

Par exemple, l'instruction suivante contient un élément `Condition` qui utilise la clé `aws:PrincipalTag` pour spécifier que le principal qui fait la requête doit être balisé avec la catégorie de tâche `iamuser-admin` .

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::111122223333:user/*",
        "Condition": {
            "StringEquals": {
                "aws:PrincipalTag/job-category": "iamuser-admin"
            }
        }
    }
}
```

Si la clé que vous spécifiez dans une condition de politique n'est pas présente dans le contexte de demande, les valeurs ne correspondent pas. Dans cet exemple, la clé `aws:PrincipalTag/job-category` est présente dans le contexte de demande si le principal utilise un utilisateur IAM avec des balises attachées. Elle est également incluse pour un principal utilisant un rôle IAM avec des balises ou des balises de session attachées. Si un utilisateur sans la balise tente d'afficher ou de modifier une clé d'accès, la condition renvoie `false` et la demande est implicitement refusée par cette instruction.

Le tableau suivant montre comment AWS évaluer cette politique en fonction des valeurs de clé de condition de votre demande.

Condition de politique	Contexte de la demande	Résultat
`"StringEquals": {`	`aws:PrincipalTag/job-category:` `  - iamuser-admin`	Correspondance

Condition de politique	Contexte de la demande	Résultat
`"aws:PrincipalTag/job-category": "iamuser-admin" }`		
`"StringEquals": { "aws:PrincipalTag/job-category": "iamuser-admin" }`	`aws:PrincipalTag/job-category: - dev-ops`	Aucune correspondance
`"StringEquals": { "aws:PrincipalTag/job-category": "iamuser-admin" }`	Aucun `aws:PrincipalTag/job-category` dans le contexte de la demande.	Aucune correspondance

Example utilisation d'une variable de politique avec une chaîne d'opérateur de condition

L'exemple suivant utilise l'opérateur de condition `StringLike` pour établir une correspondance entre une chaîne et une variable de politique afin de créer une politique qui permet à un utilisateur IAM de se servir de la console Amazon S3 pour gérer son propre « répertoire de base » dans un compartiment Amazon S3. La politique autorise les actions spécifiées dans un compartiment S3 si l'élément `s3:prefix` correspond à l'un des modèles spécifiés.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
```

```
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {
        "StringLike": {
          "s3:prefix": [
            "",
            "home/",
            "home/${aws:username}/"
          ]
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}",
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}/*"
      ]
    }
  ]
}
```

Le tableau suivant montre comment AWS évaluer cette politique pour différents utilisateurs en fonction de la aws:username valeur dans le contexte de la demande.

Condition de politique	Contexte de la requête	Résultat
`"StringLike": { "s3:prefix": [ "home/", "home/${aws:username}/" ] }`	`aws:username: - martha_rivera`	`"StringLike": { "s3:prefix": [ "home/", "home/martha_rivera /" ] }`
`"StringLike": {`	`aws:username:`	`"StringLike": {`

Condition de politique	Contexte de la requête	Résultat

<pre>"s3:prefix": [
"home/",
"home/${aws:username}/"
]
}</pre>
	<pre>- nikki_wolf</pre>	<pre>"s3:prefix": [
		"home/",
		"home/nikki_wolf /"
		]
		}</pre>

<pre>"StringLike": {
"s3:prefix": [
"home/",
"home/${aws:username}/"
]
}</pre>
	Aucun `aws:username` dans le contexte de la demande.	Aucune correspondance

Pour consulter un exemple de politique qui montre comment utiliser l'élément `Condition` pour limiter l'accès à des ressources en fonction d'un ID d'application et d'un ID utilisateur pour la fédération OIDC consultez Amazon S3 : permet aux utilisateurs Amazon Cognito d'accéder aux objets dans leur compartiment.

Opérateurs de condition de chaîne à valeurs multiples

Si une clé dans la demande contient plusieurs valeurs, les opérateurs de chaîne peuvent être qualifiés avec les opérateurs d'ensemble `ForAllValues` et `ForAnyValue`. Pour plus d'informations sur la logique d'évaluation de plusieurs clés ou valeurs de contexte, consultez Définir les opérateurs pour les clés de contexte à valeurs multiples.

Opérateur de condition	Description

`ForAllValues:StringEquals`
 	
`ForAllValues:StringEqualsIgnoreCase`
	Toutes les valeurs de la clé de condition dans la demande doivent correspondre à au moins une des valeurs de votre politique.

`ForAnyValue:StringEquals`	Au moins une valeur clé de condition dans la demande doit correspondre à l'une des valeurs de votre politique.

Opérateur de condition	Description
`ForAnyValue:StringEqualsIgnoreCase`	

`ForAllValues:StringNotEquals`
 	
`ForAllValues:StringNotEqualsIgnoreCase`
	Correspondance négative.
	
	Aucune des valeurs de la clé de contexte dans la demande ne correspond à l'une des valeurs de clé de contexte dans votre politique.

`ForAnyValue:StringNotEquals`
 	
`ForAnyValue:StringNotEqualsIgnoreCase`
	Correspondance négative.
	
	Au moins une valeur de clé de contexte dans la demande ne doit PAS correspondre à l'une des valeurs de la clé de contexte dans votre politique.

`ForAllValues:StringLike`	Toutes les valeurs de la clé de condition dans la demande doivent correspondre à au moins une des valeurs de votre politique.
`ForAnyValue:StringLike`	Au moins une valeur clé de condition dans la demande doit correspondre à l'une des valeurs de votre politique.
`ForAllValues:StringNotLike`	Correspondance négative. Aucune des valeurs de la clé de contexte dans la demande ne correspond à l'une des valeurs de clé de contexte dans votre politique.
`ForAnyValue:StringNotLike`	Correspondance négative. Au moins une valeur de clé de contexte dans la demande ne doit PAS correspondre à l'une des valeurs de la clé de contexte dans votre politique.

Example utilisation de **`ForAnyValue`** avec une chaîne d'opérateur de condition

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'utilisation de l'action `CreateTags` d'Amazon EC2 pour attacher des balises à une instance. Lorsque vous utilisez `StringEqualsIgnoreCase`, vous pouvez attacher des balises uniquement si la balise contient la clé `environment` et les valeurs `preprod` ou `storage`. Lorsque vous ajoutez `IgnoreCase` à l'opérateur, vous autorisez toute valeur de balise existante en majuscules, telle que `preprod`, `Preprod` et `PreProd`, à être résolue en vrai.

Lorsque vous ajoutez le modificateur `ForAnyValue` avec la clé de condition aws : TagKeys, au moins une valeur de clé de balise dans la demande doit correspondre à la valeur `environment`. La comparaison `ForAnyValue` est sensible à la casse, ce qui empêche les utilisateurs d'utiliser une casse incorrecte pour la clé de balise, par exemple en utilisant `Environment` au lieu de `environment`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "ec2:CreateTags",
    "Resource": "arn:aws:ec2:*:*:instance/*",
    "Condition": {
      "StringEqualsIgnoreCase": {
        "aws:RequestTag/environment": [
          "preprod",
          "storage"
        ]
      },
      "ForAnyValue:StringEquals": {
        "aws:TagKeys": "environment"
      }
    }
  }
}
```

Le tableau suivant montre comment AWS évaluer cette politique en fonction des valeurs de clé de condition de votre demande.

Condition de politique	Contexte de la requête	Résultat
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	`aws:TagKeys: - environment aws:RequestTag/environment: - preprod`	Correspondance
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	`aws:TagKeys: - environment - costcenter aws:RequestTag/environment: - PreProd`	Correspondance
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": {`	`aws:TagKeys: - Environment aws:RequestTag/Environment: - preprod`	Aucune correspondance

Condition de politique	Contexte de la requête	Résultat

<pre>"aws:TagKeys":
"environment"
}</pre>		

<pre>"StringEqualsIgnor
eCase": {
"aws:RequestTag/en
vironment": [
"preprod",
"storage"
]
},
"ForAnyValue:StringEqu
als": {
"aws:TagKeys":
"environment"
}</pre>
	<pre>aws:TagKeys:
	- costcenter
	aws:RequestTag/e
	nvironment:
	- preprod</pre>
		Aucune correspondance

<pre>"StringEqualsIgnor
eCase": {
"aws:RequestTag/en
vironment": [
"preprod",
"storage"
]
},
"ForAnyValue:StringEqu
als": {
"aws:TagKeys":
"environment"
}</pre>
	Aucun `aws:TagKeys` dans le contexte de la demande.
	<pre>aws:RequestTag/env
	ironment:
	- storage</pre>
		Aucune correspondance

Condition de politique	Contexte de la requête	Résultat
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	`aws:TagKeys: - environment` Aucun `aws:RequestTag/environment` dans le contexte de la demande.	Aucune correspondance
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	Aucun `aws:TagKeys` dans le contexte de la demande. Aucun `aws:RequestTag/environment` dans le contexte de la demande.	Aucune correspondance

Correspondance des caractères génériques

Les opérateurs de condition de chaîne réalisent une correspondance sans modèle qui n'applique pas de format prédéfini. Les opérateurs de condition ARN et de condition de date constituent un sous-ensemble d'opérateurs de chaîne qui appliquent une structure à la valeur de la clé de condition.

Nous vous recommandons d'utiliser des opérateurs de condition qui correspondent aux valeurs auxquelles vous comparez les clés. Par exemple, vous devez utiliser the section called "Opérateurs de condition de chaîne" lorsque vous comparez des clés à des valeurs de chaîne. De même, vous

devez utiliser the section called "Opérateurs de condition d'Amazon Resource Name (ARN)" lorsque vous comparez des clés à des valeurs d'ARN.

Example

Cet exemple montre comment créer une limite autour des ressources de votre organisation. La condition définie dans cette politique refuse l'accès aux actions Amazon S3, sauf si la ressource à laquelle on accède se trouve dans un ensemble spécifique d'unités organisationnelles (UO) dans AWS Organizations. Un chemin AWS Organizations est une représentation textuelle de la structure d'une entité d'organisation.

La condition exige que `aws : ResourceOrgPaths` contienne l'un des chemins d'UO répertoriés. Comme `aws:ResourceOrgPaths` est une condition à valeurs multiples, la politique utilise l'opérateur `ForAllValues:StringNotLike` pour comparer les valeurs de `aws:ResourceOrgPaths` à la liste des UO dans la politique.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyS3AccessOutsideMyBoundary",
      "Effect": "Deny",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "ForAllValues:StringNotLike": {
          "aws:ResourceOrgPaths": [
            "o-acorg/r-acroot/ou-acroot-mediaou/",
            "o-acorg/r-acroot/ou-acroot-sportsou/*"
          ]
        }
      }
    }
  ]
}
```

Le tableau suivant montre comment AWS évaluer cette politique en fonction des valeurs de clé de condition de votre demande.

Condition de politique	Contexte de la requête	Résultat
`"ForAllValues:StringNotLike": { "aws:ResourceOrgPaths": [ "o-acorg/r-acroot/ou-acroot-mediaou/", "o-acorg/r-acroot/ou-acroot-sportsou/*" ] }`	`aws:ResourceOrgPaths: - o-acorg/r-acroot/ou-acroot-sportsou/costcenter/`	Correspondance
`"ForAllValues:StringNotLike": { "aws:ResourceOrgPaths": [ "o-acorg/r-acroot/ou-acroot-mediaou/", "o-acorg/r-acroot/ou-acroot-sportsou/*" ] }`	`aws:ResourceOrgPaths: - o-acorg/r-acroot/ou-acroot-mediaou/costcenter/`	Aucune correspondance
`"ForAllValues:StringNotLike": { "aws:ResourceOrgPaths": [ "o-acorg/r-acroot/ou-acroot-mediaou/", "o-acorg/r-acroot/ou-acroot-sportsou/*" ] }`	Aucun `aws:ResourceOrgPaths:` dans la demande.	Aucune correspondance

Opérateurs de condition numériques

Les opérateurs de condition numériques permettent de créer des éléments `Condition` qui limitent l'accès après comparaison d'une clé à un entier ou une valeur décimale.

- Variables de politiques – Non prises en charge
- Caractères génériques – Non pris en charge

Opérateur de condition	Description
`NumericEquals`	Correspondance
`NumericNotEquals`	Correspondance négative
`NumericLessThan`	Correspondance « Inférieur à »
`NumericLessThanEqu als`	Correspondance « Inférieur ou égal à »
`NumericGreaterThan`	Correspondance « Supérieur à »
`NumericGreaterThan Equals`	Correspondance « Supérieur ou égal à »

Par exemple, l'instruction suivante contient un élément `Condition` qui utilise l'opérateur de condition `NumericLessThanEquals` avec la clé `s3:max-keys` pour spécifier que le demandeur peut répertorier jusqu'à 10 objets dans `amzn-s3-demo-bucket` à la fois.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
    "Condition": {"NumericLessThanEquals": {"s3:max-keys": "10"}}
  }
```

```
}
```

Si la clé que vous spécifiez dans une condition de politique n'est pas présente dans le contexte de demande, les valeurs ne correspondent pas. Dans cet exemple, la clé `s3:max-keys` est toujours présente dans la demande lorsque vous effectuez l'opération `ListBucket`. Si cette politique autorisait toutes les opérations Amazon S3, seules les opérations incluant la clé de contexte `max-keys` avec une valeur inférieure ou égale à 10 seraient autorisées.

Opérateurs de condition de date

Les opérateurs de condition de date vous permettent de créer `Condition` des éléments qui limitent l'accès en comparant une clé à une date/time valeur. Vous pouvez utiliser ces opérateurs de condition avec la clé `aws:CurrentTime` ou `aws:EpochTime`. Vous devez spécifier date/time des valeurs avec l'une des implémentations du W3C des formats de date ISO 8601 ou en heure d'époque (UNIX).

- Variables de politiques – Non prises en charge
- Caractères génériques – Non pris en charge

Opérateur de condition	Description
`DateEquals`	Correspondance à une date spécifique
`DateNotEquals`	Correspondance négative
`DateLessThan`	Correspondance avant une date et heure spécifiques
`DateLessThanEquals`	Correspondance à ou avant une date et heure spécifiques
`DateGreaterThan`	Correspondance après une date et heure spécifiques
`DateGreaterThanEqu als`	Correspondance à ou après une date et heure spécifiques

Par exemple, l'instruction suivante contient un élément `Condition` qui utilise l'opérateur de condition `DateGreaterThan` avec la clé `aws:TokenIssueTime`. Cette condition spécifie que les informations d'identification de sécurité temporaires utilisées pour effectuer la demande ont été

publiées en 2020. Cette politique peut être mise à jour par programme tous les jours pour s'assurer que les membres du compte utilisent de nouvelles informations d'identification.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::111122223333:user/*",
        "Condition": {
            "DateGreaterThan": {
                "aws:TokenIssueTime": "2020-01-01T00:00:01Z"
            }
        }
    }
}
```

Si la clé que vous spécifiez dans une condition de politique n'est pas présente dans le contexte de demande, les valeurs ne correspondent pas. La clé `aws:TokenIssueTime` n'est présente dans le contexte de demande que lorsque le principal utilise des informations d'identification temporaires pour effectuer la demande. La clé n'est pas présente dans AWS CLI les requêtes AWS d'API ou de AWS SDK effectuées à l'aide de clés d'accès. Dans cet exemple, si un utilisateur IAM tente d'afficher ou de modifier une clé d'accès, la demande est refusée.

Opérateurs de condition booléens

Les opérateurs de condition booléens permettent de créer des éléments `Condition` qui limitent l'accès après comparaison d'une clé à `true` ou `false`.

Si une clé contient plusieurs valeurs, les opérateurs booléens peuvent être qualifiés avec les opérateurs d'ensemble `ForAllValues` et `ForAnyValue`. Pour plus d'informations sur la logique d'évaluation de plusieurs clés ou valeurs de contexte, consultez Définir les opérateurs pour les clés de contexte à valeurs multiples.

- Variables de politiques – Prises en charge
- Caractères génériques – Non pris en charge

Opérateur de condition	Description
`Bool`	Correspondance booléenne
`ForAllValues:Bool`	À utiliser avec le type de données Array of Bool. Toutes les valeurs booléennes dans les valeurs clés du contexte doivent correspondre aux valeurs booléennes de votre politique. Pour empêcher les opérateurs `ForAllValues` d'évaluer les clés de contexte manquantes ou les clés de contexte avec des valeurs vides comme Autorisé, vous pouvez inclure l'opérateur de condition Null dans votre politique.
`ForAnyValue:Bool`	À utiliser avec le type de données Array of Bool. Au moins l'une des valeurs booléennes dans les valeurs clés du contexte doit correspon dre aux valeurs booléennes de votre politique.

Example opérateur de condition booléens

L politique basée sur l'identité suivante utilise l'opérateur de condition `Bool` avec la clé `aws:SecureTransport` pour refuser la réplication d'objets et de balises d'objets vers le compartiment de destination et son contenu si la demande n'est pas effectuée via SSL.

> **Important**
>
> Cette politique ne permet aucune action. Utilisez cette stratégie conjointement à d'autres stratégies qui autorisent des actions spécifiques.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "BooleanExample",
      "Action": "s3:ReplicateObject",
      "Effect": "Deny",
```

```
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket",
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ],
      "Condition": {
        "Bool": {
          "aws:SecureTransport": "false"
        }
      }
    }
  ]
}
```

Le tableau suivant montre comment AWS évaluer cette politique en fonction des valeurs de clé de condition de votre demande.

Condition de politique	Contexte de la requête	Résultat
`"Bool": { "aws:SecureTranspo rt": "false" }`	`aws:SecureTransport: - false`	Correspondance
`"Bool": { "aws:SecureTranspo rt": "false" }`	`aws:SecureTransport: - true`	Aucune correspondance
`"Bool": { "aws:SecureTranspo rt": "false" }`	Aucun `aws:Secur eTransport` dans le contexte de la demande.	Aucune correspondance

Opérateurs de condition binaires

L'opérateur de condition `BinaryEquals` vous permet de créer des éléments `Condition` qui analysent les valeurs de clé qui utilisent un format binaire. Il compare la valeur de la clé spécifiée,

octet par octet, à une représentation encodée au format Base64 de la valeur binaire dans la politique. Si la clé que vous spécifiez dans une condition de politique n'est pas présente dans le contexte de demande, les valeurs ne correspondent pas.

- Variables de politiques – Non prises en charge
- Caractères génériques – Non pris en charge

```
"Condition" : {
  "BinaryEquals": {
    "key" : "QmluYXJ5VmFsdWVJbkJhc2U2NA=="
  }
}
```

Condition de politique	Contexte de la requête	Résultat
`"BinaryEquals": { "key" : "QmluYXJ5 VmFsdWVJbkJhc2U2NA==" }`	`key: - QmluYXJ5VmFsdWVJbk Jhc2U2NA==`	Correspondance
`"BinaryEquals": { "key" : "QmluYXJ5 VmFsdWVJbkJhc2U2NA==" }`	`key: - ASIAIOSFODNN7EXAMP LE`	Aucune correspondance
`"BinaryEquals": { "key" : "QmluYXJ5 VmFsdWVJbkJhc2U2NA==" }`	Aucun `key` dans le contexte de la demande.	Aucune correspondance

Opérateurs de condition d'adresse IP

Les opérateurs de condition d'adresse IP permettent de créer des éléments `Condition` qui limitent l'accès après comparaison d'une clé à une adresse IPv4 ou IPv6 ou une plage d'adresses IP. Vous utilisez ces opérateurs avec la clé `aws:SourceIp`. La valeur doit être au format CIDR standard (par

exemple, 203.0.113). 0/24 ou 2001 DB8:1234:5678 : :/64). Si vous spécifiez une adresse IP sans préfixe de routage associé, IAM utilise la valeur de préfixe par défaut `/32`.

Certains AWS services prennent en charge IPv6, en utilisant : : pour représenter une plage de 0. Pour savoir si un service prend en charge IPv6, consultez la documentation correspondante.

- Variables de politiques – Non prises en charge
- Caractères génériques – Non pris en charge

Opérateur de condition	Description
`IpAddress`	Adresse IP ou plage d'adresses IP spécifiée
`NotIpAddress`	Toutes les adresses IP à l'exception de l'adresse IP ou de la plage d'adresse IP spécifiée

Example Opérateur de condition d'adresse IP

L'instruction suivante utilise l'opérateur de condition `IpAddress` avec la clé `aws:SourceIp` pour spécifier que la demande doit provenir d'une adresse IP comprise dans la plage 203.0.113.0 à 203.0.113.255.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::111122223333:user/*",
        "Condition": {
            "IpAddress": {
                "aws:SourceIp": "203.0.113.0/24"
            }
        }
    }
}
```

La clé de condition `aws:SourceIp` est résolue à l'aide de l'adresse IP d'où provient la demande. Si la demande provient d'une instance Amazon EC2, `aws:SourceIp` correspond à l'adresse IP publique de l'instance.

Si la clé que vous spécifiez dans une condition de politique n'est pas présente dans le contexte de demande, les valeurs ne correspondent pas. La clé `aws:SourceIp` figure toujours dans le contexte de demande, sauf lorsque le demandeur utilise un point de terminaison VPC pour effectuer la demande. Dans ce cas, la condition renvoie `false` et la demande est implicitement refusée par cette instruction.

Le tableau suivant montre comment AWS évaluer cette politique en fonction des valeurs de clé de condition de votre demande.

Condition de politique	Contexte de la requête	Résultat
`"IpAddress": { "aws:SourceIp": "203.0.113.0/24" }`	`aws:SourceIp: - 203.0.113.1`	Correspondance
`"IpAddress": { "aws:SourceIp": "203.0.113.0/24" }`	`aws:SourceIp: - 198.51.100.1`	Aucune correspondance

L'exemple suivant montre comment combiner les adresses IPv4 et IPv6 pour couvrir la totalité des adresses IP valides de l'organisation. Nous recommandons de mettre à jour les politiques de l'organisation par vos plages d'adresses IPv6 (en plus des plages d'adresses IPv4 dont vous disposez déjà) pour être sûr que les politiques continuent à fonctionner lors de la transition vers IPv6.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "someservice:*",
```

```
    "Resource": "*",
    "Condition": {
      "IpAddress": {
        "aws:SourceIp": [
          "203.0.113.0/24",
          "2001:DB8:1234:5678::/64"
        ]
      }
    }
  }
}
```

La clé de condition `aws:SourceIp` fonctionne uniquement dans une politique JSON si vous appelez l'API que vous testez directement en tant qu'utilisateur. En revanche, si vous utilisez un service pour appeler le service cible en votre nom, ce service voit l'adresse IP du service appelant plutôt que celle de l'utilisateur d'origine. Par exemple, cela peut se produire si vous utilisez AWS CloudFormation pour appeler Amazon EC2 en vue de créer automatiquement des instances. Actuellement, il n'est pas possible de transmettre l'adresse IP d'origine au service cible via un service appelant à des fins d'évaluation dans une politique JSON. Pour ces types d'appels d'API de service, vous ne devez pas utiliser la clé de condition `aws:SourceIp`.

Opérateurs de condition d'Amazon Resource Name (ARN)

Les opérateurs de condition d'Amazon Resource Name (ARN) permettent de créer des éléments `Condition` qui limitent l'accès après comparaison d'une clé à un ARN. L'ARN est considéré comme étant une chaîne.

- Variables de politiques – Prises en charge
- Caractères génériques – Pris en charge

Opérateur de condition	Description
`ArnEquals` , `ArnLike`	Case-sensitive correspondance de l'ARN. Chacun des six composant s de l'ARN, séparés par deux points, est vérifié séparément et chacun peut inclure un caractère générique correspondant à plusieurs caractères (*) ou un caractère générique correspondant à un caractère (?). Les opérateurs de condition `ArnEquals` et `ArnLike` se comportent de manière identique.

Opérateur de condition	Description
`ArnNotEquals` , `ArnNotLike`	Correspondance négative à l'ARN. Les opérateurs de condition `ArnNotEquals` et `ArnNotLike` se comportent de manière identique.

Example Opérateur de condition de l'ARN

L'exemple de politique basée sur les ressources suivant montre une politique attachée à une file d'attente Amazon SQS à laquelle vous souhaitez envoyer des messages SNS. Elle donne l'autorisation à Amazon SNS d'envoyer des messages à une ou plusieurs files d'attente de votre choix, mais uniquement si le service envoie le message pour le compte d'une ou plusieurs rubriques Amazon SNS spécifiques. Vous spécifiez la file d'attente dans le champ `Resource`, tandis que la rubrique Amazon SNS est la valeur de la clé `SourceArn`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {
            "Service": "sns.amazonaws.com"
        },
        "Action": "SQS:SendMessage",
        "Resource": "arn:aws:sqs:us-east-1:123456789012:QUEUE-ID",
        "Condition": {
            "ArnEquals": {
                "aws:SourceArn": "arn:aws:sns:us-east-1:123456789012:TOPIC-ID"
            }
        }
    }
}
```

La clé `aws:SourceArn` figure dans le contexte de demande uniquement si une ressource déclenche un service pour appeler un autre service au nom du propriétaire de la ressource. Si un utilisateur IAM tente d'effectuer cette opération directement, la condition renvoie `false` et la demande est implicitement refusée par cette instruction.

Le tableau suivant montre comment AWS évaluer cette politique en fonction des valeurs de clé de condition de votre demande.

Condition de politique	Contexte de la requête	Résultat
`"ArnEquals": { "aws:SourceArn": "arn:aws:sns:us-west-2:123456789012:TOPIC-ID" }`	`aws:SourceArn: - arn:aws:sns:us-west-2:123456789012:TOPIC-ID`	Correspondance
`"ArnEquals": { "aws:SourceArn": "arn:aws:sns:us-west-2:123456789012:TOPIC-ID" }`	`aws:SourceArn: - arn:aws:sns:us-west-2:777788889999:TOPIC-ID`	Aucune correspondance
`"ArnEquals": { "aws:SourceArn": "arn:aws:sns:us-west-2:123456789012:TOPIC-ID" }`	Aucun `aws:SourceArn` dans le contexte de la demande.	Aucune correspondance

Opérateurs de condition d'ARN à valeurs multiples

Si une clé dans la demande contient plusieurs valeurs, les opérateurs d'ARN peuvent être qualifiés avec les opérateurs d'ensemble `ForAllValues` et `ForAnyValue`. Pour plus d'informations sur la logique d'évaluation de plusieurs clés ou valeurs de contexte, consultez Définir les opérateurs pour les clés de contexte à valeurs multiples.

Opérateur de condition	Description

`ForAllValues:ArnEquals`
 	
`ForAllValues:ArnLike`
	Tous les ARN du contexte de la demande doivent correspondre à au moins un des modèles ARN de votre politique.

`ForAnyValue:ArnEquals`
 	
`ForAnyValue:ArnLike`
	Au moins un ARN du contexte de la demande doit correspondre à un des modèles ARN de votre politique.

`ForAllValues:ArnNotEquals`
 	
`ForAllValues:ArnNotLike`
	Correspondance négative.
	
	Aucun des ARN dans le contexte de la demande ne correspond à un modèle ARN de chaîne dans votre politique.

`ForAnyValue:ArnNotEquals`
 	
`ForAnyValue:ArnNotLike`
	Correspondance négative.
	
	Au moins un ARN dans le contexte de la demande ne doit PAS correspondre à l'un des modèles ARN de votre politique.

Example utilisation de **`ForAllValues`** avec un d'opérateur de condition d'ARN

L'exemple suivant permet `ForAllValues:ArnLike` de créer ou de mettre à jour une source de diffusion logique pour les CloudWatch journaux Amazon Logs. Le bloc de condition inclut la clé de condition `logs:LogGeneratingResourceArns` pour filtrer les ARN de ressources générant des journaux transmis dans la demande. À l'aide de cet opérateur de condition, tous les ARN de la demande doivent correspondre à au moins un ARN de la politique.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
```

```
            "Action": "logs:PutDeliverySource",
            "Resource": "arn:aws:logs:us-east-1:123456789012:delivery-source:*",
            "Condition": {
                "ForAllValues:ArnLike": {
                    "logs:LogGeneratingResourceArns": [
                        "arn:aws:cloudfront::123456789012:distribution/*",
                        "arn:aws:cloudfront::123456789012:distribution/support*"
                    ]
                }
            }
        }
    ]
}
```

Le tableau suivant montre comment AWS évaluer cette politique en fonction des valeurs de clé de condition de votre demande.

Condition de politique	Contexte de la requête	Résultat
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*" ] }`	`logs:LogGeneratingResourceArns: - arn:aws::cloudfront:123456789012:distribution/costcenter`	Correspondance
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*",`	`logs:LogGeneratingResourceArns: - arn:aws::cloudfront:123456789012:distribution/costcenter - arn:aws::cloudfront:123456789012:distribution/support2025`	Correspondance

Condition de politique	Contexte de la requête	Résultat
`"arn:aws::cloudfront:123456789012:distribution/support*" ] }`		
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*" ] }`	`logs:LogGeneratingResourceArns: - arn:aws::cloudfront:123456789012:distribution/costcenter - arn:aws::cloudfront:123456789012:distribution/admin`	Correspondance
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*" ] }`	`logs:LogGeneratingResourceArns: - arn:aws::cloudfront:777788889999:distribution/costcenter`	Aucune correspondance

Condition de politique	Contexte de la requête	Résultat
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*" ] }`	Aucun `logs:LogGeneratingResourceArns` dans le contexte de la demande.	Correspondance

Le qualificateur `ForAllValues` renvoie la valeur true si la demande ne comprend pas de clés de contexte ou si la valeur de clé de contexte aboutit à un jeu de données nul, tel qu'une chaîne vide. Pour éviter que des clés de contexte manquantes ou des clés de contexte contenant des valeurs vides ne soient considérées comme vraies, vous pouvez inclure l'opérateur de condition Null dans votre politique avec une valeur `false` pour vérifier si la clé de contexte existe et si sa valeur n'est pas nulle.

... IfExists opérateurs conditionnelles

Vous pouvez ajouter `IfExists` à la fin de n'importe quel nom d'opérateur de condition, à l'exception de la condition `Null`, par exemple,`StringLikeIfExists`. Ceci équivaut à spécifier que « Si la clé de condition est présente dans le contexte de la requête, la clé doit être traitée comme spécifié dans la politique. Si la clé n'est pas présente, la condition évalue l'élément de condition comme vrai. » Les autres éléments de condition dans l'instruction peuvent toujours se traduire par une absence de correspondance, mais pas par une clé manquante lors de la vérification avec `...IfExists`. Si vous utilisez un élément `"Effect": "Deny"` avec un opérateur de condition négatif tel que `StringNotEqualsIfExists`, la demande est toujours refusée, même si la clé de condition n'est pas présente.

Exemple d'utilisation de **`IfExists`**

De nombreuses clés de condition contiennent des informations se rapportant à un type spécifique de ressources et elles ne sont présentes que lorsque vous accédez à ce type de ressources. Ces clés de condition n'existent pas pour les autres types de ressources. Le fait que l'instruction ne s'applique

qu'à un type spécifique de ressources ne pose pas problème. Toutefois, dans certains scénarios, une même instruction peut s'appliquer à plusieurs types de ressources, par exemple lorsque l'instruction de politique référence les actions de plusieurs services ou lorsqu'une action donnée d'un service accède à différents types de ressources dans un même service. Dans ce cas, l'inclusion d'une clé de condition applicable uniquement à une des ressources dans l'instruction de politique peut provoquer l'échec de l'élément `Condition` et de ce fait, l'élément `"Effect"` ne s'applique pas.

Prenons l'exemple de politique suivant :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "THISPOLICYDOESNOTWORK",
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": "*",
    "Condition": {"StringLike": {"ec2:InstanceType": [
      "t1.*",
      "t2.*",
      "m3.*"
    ]}}
  }
}
```

L'intention de la politique précédente est de permettre à l'utilisateur de lancer n'importe quelle instance de type `t1`, `t2` ou `m3`. Toutefois, le lancement d'une instance requiert non seulement l'accès à l'instance proprement dite, mais également à de nombreuses ressources telles que des images, des paires de clés, des groupes de sécurité, etc. L'ensemble de l'instruction est évalué par rapport à chaque ressource requise pour le lancement de l'instance. Ces ressources supplémentaires n'ont pas la clé de condition `ec2:InstanceType` et, par conséquent, la vérification `StringLike` échoue et l'utilisateur n'est autorisé à lancer aucun type d'instance.

Pour éviter ce problème, utilisez l'opérateur de condition `StringLikeIfExists` à la place. De cette façon, le test n'est effectué que si la clé de condition existe. Cela peut être interprété comme suit : « Si la ressource en cours de vérification est dotée d'une clé de condition `ec2:InstanceType`, l'action peut uniquement être autorisée si la valeur de la clé commence par `t1.`, `t2.` ou `m3.`. Si la ressource en cours de vérification n'est pas dotée de cette clé de condition, peu importe. »

L'astérisque (*) figurant dans les valeurs des clés de condition, lorsqu'il est utilisé avec l'opérateur de condition `StringLikeIfExists`, est interprété comme un caractère générique pour obtenir des correspondances de chaînes partielles. L'instruction `DescribeActions` inclut les actions requises pour afficher l'instance dans la console.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RunInstance",
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*",
      "Condition": {
        "StringLikeIfExists": {
          "ec2:InstanceType": [
            "t1.*",
            "t2.*",
            "m3.*"
          ]
        }
      }
    },
    {
      "Sid": "DescribeActions",
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeImages",
        "ec2:DescribeInstances",
        "ec2:DescribeVpcs",
        "ec2:DescribeKeyPairs",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups"
      ],
      "Resource": "*"
    }
  ]
}
```

Le tableau suivant montre comment AWS évaluer cette politique en fonction des valeurs de clé de condition de votre demande.

Condition de politique	Contexte de la requête	Résultat
`"StringLikeIfExists": { "ec2:InstanceType": [ "t1.*", "t2.*", "m3.*" ] }`	`ec2:InstanceType: - t1.micro`	Correspondance
`"StringLikeIfExists": { "ec2:InstanceType": [ "t1.*", "t2.*", "m3.*" ] }`	`ec2:InstanceType: - m2.micro`	Aucune correspondance
`"StringLikeIfExists": { "ec2:InstanceType": [ "t1.*", "t2.*", "m3.*" ] }`	Aucun `ec2:InstanceType` dans le contexte de la demande.	Correspondance

Opérateur de condition pour vérifier l'existence de clés de condition

Utilisez un opérateur de condition `Null` pour vérifier si une clé de condition est absente au moment de l'autorisation. Dans l'instruction de politique, utilisez `true` (la clé n'existe pas ; elle est nulle) ou `false` (la clé existe et sa valeur n'est pas nulle).

Vous ne pouvez pas utiliser une variable de politique avec l'opérateur de condition `Null`.

Par exemple, vous pouvez utiliser cet opérateur de condition pour déterminer si un utilisateur utilise des informations d'identification temporaires ou ses propres informations d'identification pour effectuer une demande. S'il utilise des informations d'identification temporaires, la clé `aws:TokenIssueTime` existe et elle est dotée d'une valeur. L'exemple suivant illustre une condition qui spécifie que l'utilisateur doit avoir recours à des informations d'identification temporaires (la clé ne peut pas être absente) pour exécuter l'API Amazon EC2.

JSON

```
{
  "Version":"2012-10-17",
  "Statement":{
      "Action":"ec2:*",
      "Effect":"Allow",
      "Resource":"*",
      "Condition":{"Null":{"aws:TokenIssueTime":"false"}}
  }
}
```

Conditions avec plusieurs clés ou valeurs de contexte

Vous pouvez utiliser l'élément `Condition` d'une politique pour tester plusieurs clés ou plusieurs valeurs de contexte pour une seule clé de contexte dans une demande. Lorsque vous faites une demande AWS, par programmation ou par le biais du Console de gestion AWS, votre demande inclut des informations sur votre principal, votre fonctionnement, vos balises, etc. Vous pouvez utiliser des clés de contexte pour tester les valeurs des clés de contexte correspondantes dans la demande, avec les clés de contexte spécifiées dans la condition de politique. Pour en savoir plus sur les informations et les données incluses dans une demande, consultez Contexte de la demande.

Rubriques

- Logique d'évaluation pour plusieurs clés ou valeurs de contexte

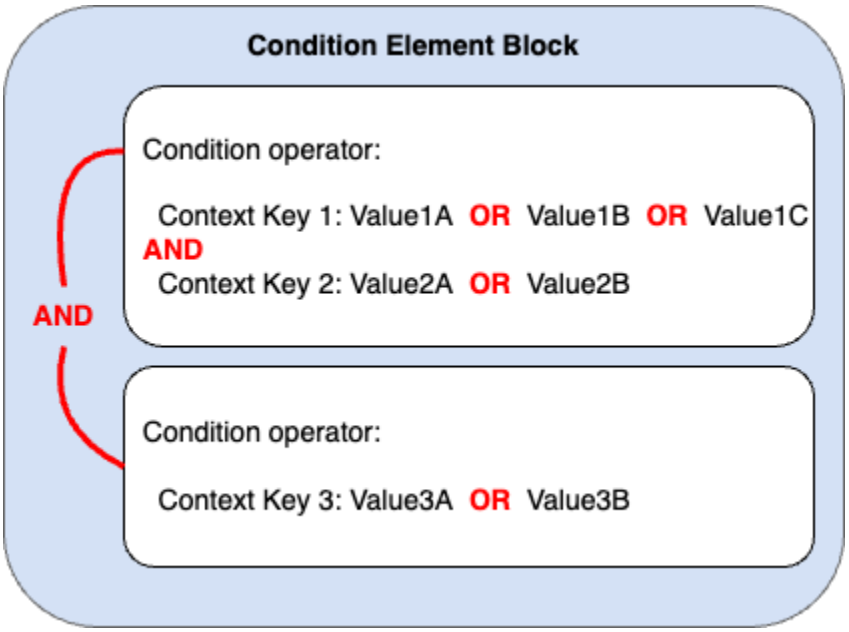
- Logique d'évaluation des opérateurs de condition de correspondance négative

Logique d'évaluation pour plusieurs clés ou valeurs de contexte

Un élément `Condition` peut contenir plusieurs opérateurs de condition, et chaque opérateur de condition peut également inclure plusieurs paires clé-valeur de contexte. La plupart des clés de contexte prennent en charge l'utilisation de plusieurs valeurs, sauf indication contraire.

- Si votre instruction de politique dispose de plusieurs opérateurs de condition, ils sont évalués à l'aide d'un opérateur logique `AND`.
- Si votre instruction de politique dispose de plusieurs clés de contexte attachées à un même opérateur de condition, celles-ci sont évaluées à l'aide d'un opérateur logique `AND`.
- Si un même opérateur de condition comprend plusieurs valeurs pour une clé de contexte, ces valeurs sont évaluées à l'aide d'un opérateur logique `OR`.
- Si un même opérateur de condition de correspondance négative comprend plusieurs valeurs pour une clé de contexte, ces valeurs sont évaluées à l'aide d'un opérateur logique `NOR`.

Toutes les clés de contexte d'un bloc d'éléments de condition doivent être résolues sur true (vrai) pour invoquer l'effet `Allow` ou `Deny` souhaité. La figure suivante illustre la logique d'évaluation d'une condition comportant plusieurs opérateurs de condition et des paires clé-valeur de contexte.



Par exemple, la politique de compartiment S3 suivante illustre la manière dont la figure précédente est représentée dans une politique. Le bloc de conditions comprend les opérateurs de condition

`StringEquals` et `ArnLike`, et les clés de contexte `aws:PrincipalTag` et `aws:PrincipalArn`. Afin d'invoquer l'effet `Allow` ou `Deny` souhaité, toutes les clés de contexte du bloc de conditions doivent être résolues sur true (vrai). L'utilisateur qui fait la demande doit disposer des deux clés de balise du principal, department et role, qui incluent l'une des valeurs de clés de balise spécifiées dans la politique. De plus, l'ARN principal de l'utilisateur qui fait la demande doit correspondre à l'une des valeurs `aws:PrincipalArn` spécifiées dans la politique pour être évalué comme true (vrai).

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": [
            "finance",
            "hr",
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
```

```
}
```

Le tableau suivant montre comment cette politique est AWS évaluée en fonction des valeurs des clés de condition figurant dans votre demande.

Condition de politique	Contexte de la demande	Résultat
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	`aws:PrincipalTag/department: legal aws:PrincipalTag/role: audit aws:PrincipalArn: arn:aws:iam::222222222222:user/Mary`	Match
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ],`	`aws:PrincipalTag/department: hr aws:PrincipalTag/role: audit aws:PrincipalArn: arn:aws:iam::222222222222:user/ Nikki`	Aucune correspondance

Condition de politique	Contexte de la demande	Résultat

<pre>"aws:PrincipalTag/role": [
"audit",
"security"
]
},
"ArnLike": {
"aws:PrincipalArn": [
"arn:aws:iam::222222222222:user/Ana",
"arn:aws:iam::222222222222:user/Mary"
]
}</pre>		

Condition de politique	Contexte de la demande	Résultat
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	`aws:PrincipalTag/department: hr aws:PrincipalTag/role: payroll aws:PrincipalArn: arn:aws:iam::22222222222222:user/Mary`	Aucune correspondance

Condition de politique	Contexte de la demande	Résultat

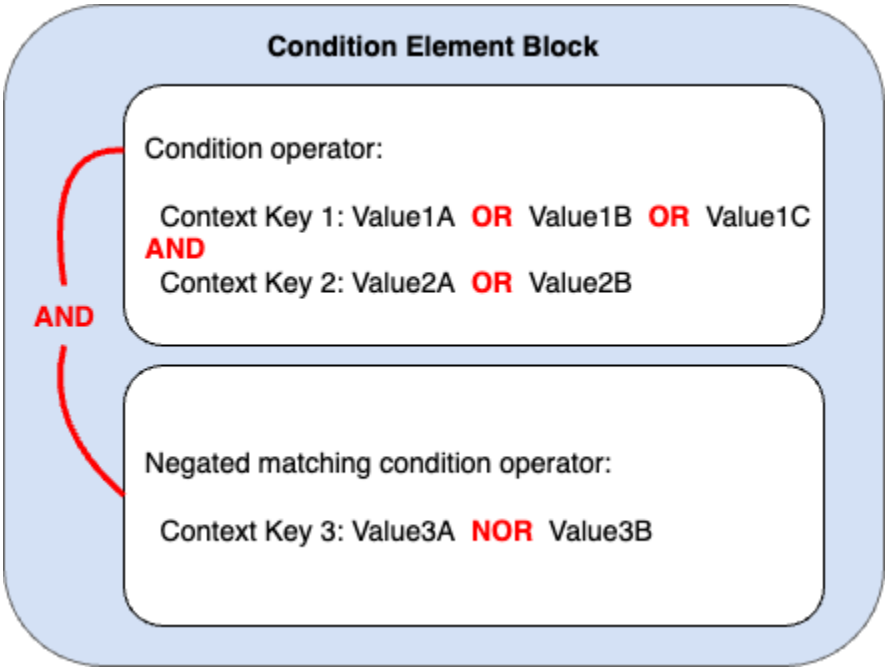
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	Aucun `aws:PrincipalTag/role` dans le contexte de la demande.
	`aws:PrincipalTag/department: hr`
	`aws:PrincipalArn: arn:aws:iam::222222222222:user/Mary`
		Aucune correspondance

Condition de politique	Contexte de la demande	Résultat
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	Aucun `aws:PrincipalTag` dans le contexte de la demande. `aws:PrincipalArn: arn:aws:iam::22222 2222222:user/Mary`	Aucune correspondance

Logique d'évaluation des opérateurs de condition de correspondance négative

Certains opérateurs de condition, tels que `StringNotEquals` ou `ArnNotLike`, utilisent la correspondance négative pour comparer les paires clé-valeur de contexte de votre politique aux paires clé-valeur de contexte d'une demande. Lorsque plusieurs valeurs sont spécifiées pour une même clé de contexte dans une politique avec des opérateurs de condition de correspondance négative, les autorisations effectives fonctionnent comme un opérateur logique `NOR`. Dans le cas d'une correspondance négative, un opérateur logique `NOR` ou `NOT OR` renvoie la valeur true (vrai) uniquement si toutes les valeurs sont évaluées comme false (fausses).

La figure suivante illustre la logique d'évaluation d'une condition comportant plusieurs opérateurs de condition et des paires clé-valeur de contexte. La figure inclut un opérateur de condition de correspondance négative pour la clé de contexte 3.



Par exemple, la politique de compartiment S3 suivante illustre la manière dont la figure précédente est représentée dans une politique. Le bloc de conditions comprend les opérateurs de condition `StringEquals` et `ArnNotLike`, et les clés de contexte `aws:PrincipalTag` et `aws:PrincipalArn`. Afin d'invoquer l'effet `Allow` ou `Deny` souhaité, toutes les clés de contexte du bloc de conditions doivent être résolues sur true (vrai). L'utilisateur qui fait la demande doit disposer des deux clés de balise du principal, department et role, qui incluent l'une des valeurs de clés de balise spécifiées dans la politique. Puisque l'opérateur de condition `ArnNotLike` utilise la correspondance négative, l'ARN principal de l'utilisateur qui fait la demande ne doit pas correspondre à l'une des valeurs `aws:PrincipalArn` spécifiées dans la politique pour être évalué comme true (vrai).

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
```

```
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": [
            "finance",
            "hr",
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnNotLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
}
```

Le tableau suivant montre comment cette politique est AWS évaluée en fonction des valeurs des clés de condition figurant dans votre demande.

Condition de politique	Contexte de la demande	Résultat
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ]`	`aws:PrincipalTag/department: legal aws:PrincipalTag/role: audit aws:PrincipalArn: arn:aws:iam::222222222222:user/Nikki`	Match

Condition de politique	Contexte de la demande	Résultat
}, "ArnNotLike": { "aws:PrincipalArn": ["arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary"] }		
"StringEquals": { "aws:PrincipalTag/department": ["finance", "hr", "legal"], "aws:PrincipalTag/role": ["audit", "security"] }, "ArnNotLike": { "aws:PrincipalArn": ["arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary"] }	aws:PrincipalTag/department: hr aws:PrincipalTag/role: audit aws:PrincipalArn: arn:aws:iam::222222222222:user/ *Mary*	Aucune correspondance

Condition de politique	Contexte de la demande	Résultat
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnNotLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	`aws:PrincipalTag/department: hr aws:PrincipalTag/role: payroll aws:PrincipalArn: arn:aws:iam::222222222222:user/Nikki`	Aucune correspondance

Condition de politique	Contexte de la demande	Résultat
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnNotLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	**>** Aucun `aws:PrincipalTag/role` dans le contexte de la demande. `aws:PrincipalTag/department: hr aws:PrincipalArn: arn:aws:iam::222222222222:user/Nikki`	Aucune correspondance

Condition de politique	Contexte de la demande	Résultat
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnNotLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	Aucun `aws:PrincipalTag` dans le contexte de la demande. `aws:PrincipalArn: arn:aws:iam::22222 2222222:user/Nikki`	Aucune correspondance

Single-valued par rapport aux clés de contexte à valeurs multiples

La différence entre les clés de contexte à valeur unique et à valeurs multiples réside dans le nombre de valeurs dans le contexte de la requête, et non du nombre de valeurs dans la condition de la politique.

- Single-valuedles clés de contexte de condition ont au plus une valeur dans le contexte de la demande. Par exemple, lorsque vous balisez des ressources dans AWS, chaque étiquette de ressource est stockée sous la forme d'une paire clé-valeur. Comme une clé de balise de ressource ne peut avoir qu'une seule valeur de balise, the section called "ResourceTag" est une clé de contexte à valeur unique. N'utilisez pas d'opérateur d'ensemble de conditions avec une clé de contexte à valeur unique.

- Les clés de contexte de condition à valeurs multiples peuvent comporter plusieurs valeurs dans le contexte de la demande. Par exemple, lorsque vous balisez des ressources dans AWS, vous pouvez inclure plusieurs paires clé-valeur de balise dans une seule demande. Ainsi, the section called "TagKeys" est une clé de contexte à valeurs multiples. Les clés de contexte à valeurs multiples nécessitent un opérateur d'ensemble de conditions.

Par exemple, une demande peut provenir d'un seul point de terminaison d'un VPC au maximum. the section called "SourceVpce" est donc une clé de contexte à valeur unique. Étant donné qu'un service peut avoir plus d'un nom de principal de service qui appartient au service, the section called "PrincipalServiceNamesList" est une clé de contexte à valeurs multiples.

> **Important**
>
> La différence entre les clés de contexte à valeur unique et à valeurs multiples dépend du nombre de valeurs dans le contexte de la demande, et non du nombre de valeurs dans la condition de la politique.

Pour des exemples de politiques illustrant des conditions avec plusieurs clés de contexte et valeurs, consultezExemples de politiques de conditions.

Points clés

- Les classifications Single-valued et les classifications à valeurs multiples sont incluses dans la description de chaque clé de contexte de condition en tant que type de valeur dans la AWS clés de contexte de condition globale rubrique.
- Les clés de contexte à valeurs multiples dans la référence d'autorisation de service utilisent un préfixe `ArrayOf` suivi du type d'opérateur de condition, tel que `ArrayOfString` ou `ArrayOfARN`, indiquant que la demande peut inclure plusieurs valeurs pour une clé de contexte de condition.
- Vous pouvez utiliser n'importe quelle clé de contexte à valeur unique disponible en tant que variable de politique, mais vous ne pouvez pas utiliser une clé de contexte à valeurs multiples en tant que variable de politique. Pour de plus amples informations sur les variables de politique, veuillez consulter Éléments des politiques IAM : variables et balises.
- Lors de l'utilisation de clés de contexte comprenant des paires clé-valeur, il est important de noter que même s'il peut y avoir plusieurs valeurs de clé de balise, chacun `tag-key` ne peut avoir qu'une seule valeur.

- the section called "PrincipalTag", the section called "RequestTag" et the section called "ResourceTag" sont des clés de contexte à valeur unique.
- the section called "TagKeys" définit les clés de balise autorisées dans une demande, mais n'inclut pas les valeurs des clés de balise. Comme vous pouvez inclure plusieurs paires clé-valeur dans une demande, `aws:TagKeys` est une clé de contexte à valeurs multiples.

- Les clés de contexte à valeurs multiples nécessitent un opérateur d'ensemble de conditions. N'utilisez pas d'opérateurs d'ensemble de conditions `ForAllValues` ou `ForAnyValue` avec des clés de contexte à valeur unique. L'utilisation d'opérateurs d'ensembles de conditions avec des clés de contexte à valeur unique peut entraîner des politiques trop permissives.

Définir les opérateurs pour les clés de contexte à valeurs multiples

Pour comparer votre clé de contexte de condition à un contexte de demande à valeurs multiples, vous devez utiliser les opérateurs d'ensemble `ForAllValues` ou `ForAnyValue`. Ces opérateurs d'ensembles sont utilisés pour comparer deux ensembles de valeurs, tels que l'ensemble de balises dans une demande et l'ensemble de balises dans une condition de politique.

Ces qualificatifs `ForAllValues` et `ForAnyValue` ajoutent une fonctionnalité d'opération d'ensemble à l'opérateur de condition afin que vous puissiez tester les clés de contexte avec plusieurs valeurs par rapport à plusieurs valeurs de clé de contexte dans une condition de politique. En outre, si vous incluez une clé de contexte à valeurs multiples dans votre politique avec un caractère générique ou une variable, vous devez également utiliser l'opérateur de condition `StringLike`. Les valeurs de clé de condition multiples doivent être placées entre crochets, comme dans un tableau, par exemple, `"Key2":["Value2A", "Value2B"]`.

ForAllValues

Le qualificateur `ForAllValues` vérifie si la valeur de chaque membre du contexte de la demande correspond à l'opérateur de condition qui suit le qualificateur. La condition renvoie la valeur `true` si chaque valeur de clé de contexte de la demande correspond à une valeur de clé de contexte de la politique. Il renvoie également `true` s'il n'y a aucune clé de contexte dans la demande.

> **Important**
>
> Faites preuve de vigilance si vous utilisez `ForAllValues` avec un effet `Allow`, car cela peut être trop permissif si la présence de clés de contexte manquantes dans le contexte de la demande est inattendue. Vous devez toujours inclure l'opérateur de condition `Null` dans

votre politique avec une valeur `false` pour vérifier si la clé de contexte existe et si sa valeur n'est pas nulle. Pour obtenir un exemple, consultez Contrôle de l'accès en fonction des clés de balise.

Exemple d'opérateur ForAllValues d'ensemble

Dans l'exemple suivant, ForAllValues est utilisé avec aws : TagKeys pour permettre aux utilisateurs de supprimer des balises spécifiques attribuées à une instance EC2. Cette politique permet aux utilisateurs de supprimer uniquement les balises `environment` et `cost-center`. Vous pouvez les supprimer séparément ou ensemble. Les clés de balise dans la demande doivent correspondre exactement aux clés spécifiées dans la politique.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:DeleteTags",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "environment",
                        "cost-center"
                    ]
                },
                "Null": {
                    "aws:TagKeys": "false"
                }
            }
        }
    ]
}
```

Le tableau suivant montre comment AWS évaluer cette politique en fonction des valeurs de clé de condition de votre demande.

Condition de politique	Contexte de la demande	Résultat
`"ForAllValues:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }, "Null": { "aws:TagKeys": "false" }`	`aws:TagKeys: - environment`	Match
`"ForAllValues:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }, "Null": { "aws:TagKeys": "false" }`	`aws:TagKeys: - cost-center`	Match
`"ForAllValues:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }, "Null": { "aws:TagKeys": "false" }`	`aws:TagKeys: - environment - cost-center`	Match
`"ForAllValues:StringEquals": {`	`aws:TagKeys: - environment`	Aucune correspondance

Condition de politique	Contexte de la demande	Résultat

<pre>"aws:TagKeys": [
"environment",
"cost-center"
]
},
"Null": {
"aws:TagKeys":
"false"
}</pre>
	<pre>- dept</pre>	

<pre>"ForAllValues:Stri
ngEquals": {
"aws:TagKeys": [
"environment",
"cost-center"
]
},
"Null": {
"aws:TagKeys":
"false"
}</pre>
	Aucun `aws:TagKeys` dans le contexte de la demande.	Aucune correspondance

Notez que dans le dernier exemple, le résultat est « Aucune correspondance » car la vérification de la condition Null empêche la correspondance lorsque la clé de contexte est manquante. Une bonne pratique consiste à éviter les politiques trop permissives.

ForAnyValue

Le qualificateur `ForAnyValue` teste si au moins un membre de l'ensemble des valeurs de clé de contexte de la demande correspond à au moins un membre de l'ensemble des valeurs de clé de contexte de votre condition de politique. La condition renvoie la valeur `true` si l'une des valeurs de clé de contexte de la demande correspond à l'une des valeurs de clé de contexte de la politique. Si aucune clé de contexte correspondante n'est trouvée ou si la clé n'existe pas, la condition renvoie `false`.

> **Important**
>
> Lorsque vous utilisez `ForAnyValue` avec un effet `Deny`, si la clé de contexte n'est pas présente dans la demande, la politique est évaluée comme Aucune correspondance. Pour garantir un comportement cohérent, ajoutez une vérification de condition `Null` explicite dans votre politique afin de vérifier si la clé de contexte existe. Pour en savoir plus, consultez Opérateur de condition pour vérifier l'existence de clés de condition.

Exemple d'opérateur ForAnyValue d'ensemble

Dans l'exemple suivant, ForAnyValue est utilisé avec aws : TagKeys pour permettre aux utilisateurs de supprimer des balises spécifiques attribuées à une instance EC2. Cette politique permet aux utilisateurs de supprimer les balises pour une instance si les clés de balise spécifiées dans la demande incluent `environment` ou `cost-center`. La demande peut inclure des clés de balise supplémentaires en plus de celles spécifiées dans la politique, mais doit inclure au moins une des clés spécifiées pour correspondre à la condition.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:DeleteTags",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "environment",
                        "cost-center"
                    ]
                }
            }
        }
    ]
}
```

Le tableau suivant montre comment AWS évaluer cette politique en fonction des valeurs de clé de condition de votre demande.

Condition de politique	Contexte de la demande	Résultat
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - environment`	Match
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - cost-center`	Match
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - environment - cost-center`	Match
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - environment - dept`	Match
`"ForAnyValue:StringEquals": {`	`aws:TagKeys:`	Aucune correspondance

Condition de politique	Contexte de la demande	Résultat
`"aws:TagKeys": [ "environment", "cost-center" ] }`	`- dept`	
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	Aucun `aws:TagKeys` dans le contexte de la demande.	Aucune correspondance

Exemples de politiques de conditions

Dans les politiques IAM, vous pouvez spécifier plusieurs valeurs pour les clés de contexte à valeur unique et à valeurs multiples à des fins de comparaison avec le contexte de la demande. L'ensemble suivant d'exemples de politiques montre les conditions de politique avec plusieurs clés et valeurs de contexte.

> **Note**
>
> Si vous souhaitez envoyer une politique à inclure dans ce guide de référence, utilisez le bouton Commentaire situé au bas de cette page. Pour accéder à des exemples de politiques IAM basées sur l'identité, veuillez consulter la rubrique Exemples de politiques basées sur l'identité IAM.

Exemples de politiques conditionnelles : clés Single-valued contextuelles

- Blocs de conditions multiples avec clés de contexte à valeur unique
- Un bloc de conditions avec plusieurs clés de contexte et valeurs à valeur unique

Exemples de politiques de conditions : clés de contexte à valeurs multiples.

- Politique de refus avec opérateur de condition ForAllValues
- Politique de refus avec opérateur de condition ForAnyValue

Exemples de clés de contexte à valeurs multiples

L'ensemble suivant d'exemples de politiques montre comment créer des conditions de politique à l'aide de clés de contexte à valeurs multiples.

Exemple : politique de refus avec opérateur de jeu de conditions ForAllValues

Les exemples suivants montrent comment utiliser une politique basée sur l'identité pour interdire l'utilisation d'actions de balisage IAM lorsque des préfixes de clé de balise spécifiques sont inclus dans la demande. Les valeurs pour `aws:TagKeys` incluent un caractère générique (*) pour la correspondance partielle des chaînes. La politique inclut l'opérateur d'ensemble `ForAllValues` avec la clé de contexte `aws:TagKeys`, car la clé de contexte de la demande peut inclure plusieurs valeurs. Afin que la clé de contexte `aws:TagKeys` corresponde, chaque valeur du contexte de la demande doit correspondre à au moins une valeur de la politique.

L'opérateur d'ensemble `ForAllValues` renvoie également vrai s'il n'y a aucune clé de contexte dans la demande.

Vous pouvez éviter que des clés de contexte manquantes ou des clés de contexte contenant des valeurs vides ne soient considérées comme vraies en incluant l'opérateur de condition `Null` dans votre politique avec une valeur `false` pour vérifier si la clé de contexte de la demande existe et si sa valeur n'est pas nulle. Pour de plus amples informations, veuillez consulter Opérateur de condition pour vérifier l'existence de clés de condition.

> **Important**
>
> Cette politique ne permet aucune action. Utilisez cette stratégie conjointement à d'autres stratégies qui autorisent des actions spécifiques.

Example Refuser une seule valeur de condition de politique pour une clé de contexte à valeurs multiples

Dans l'exemple suivant, la politique refuse les demandes dont les valeurs pour `aws:TagKeys` dans la demande n'incluent pas le préfixe key1. Le contexte de demande peut comporter plusieurs valeurs,

mais en raison de l'opérateur d'ensemble de conditions `ForAllValues`, toutes les valeurs de clé de balise du contexte de demande doivent commencer par le préfixe key1.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyRestrictedTags",
      "Effect": "Deny",
      "Action": [
        "iam:Tag*",
        "iam:UnTag*"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "ForAllValues:StringNotLike": {
          "aws:TagKeys": "key1*"
        }
      }
    }
  ]
}
```

Le tableau suivant montre comment cette politique est AWS évaluée en fonction des valeurs des clés de condition figurant dans votre demande. Pour une instruction Refuser, Correspondance est refusée et Aucune correspondance n'est pas refusée, elle peut donc être autorisée par une autre instruction.

Condition de politique	Contexte de la demande	Résultat
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	`aws:TagKeys: - key1:legal`	Aucune correspondance Peut être autorisé par une autre instruction.

Condition de politique	Contexte de la demande	Résultat
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	`aws:TagKeys: - key1:hr - key1:personnel`	Aucune correspondance Peut être autorisé par une autre instruction.
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	`aws:TagKeys: - key2:audit`	Match
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	Aucun `aws:TagKeys` dans le contexte de la demande.	Match

Example Refuser plusieurs valeurs de condition de politique pour une clé de contexte à valeurs multiples

Dans l'exemple suivant, la politique refuse les demandes dont les valeurs pour `aws:TagKeys` dans la demande n'incluent pas le préfixe key1 ou key2. Le contexte de demande peut comporter plusieurs valeurs, mais en raison de l'opérateur d'ensemble de conditions `ForAllValues`, toutes les valeurs de clé de balise du contexte de demande doivent commencer par le préfixe key1 ou key2.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyRestrictedTags",
      "Effect": "Deny",
      "Action": [
        "iam:Tag*",
```

```
        "iam:UnTag*"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "ForAllValues:StringNotLike": {
          "aws:TagKeys": [
            "key1*",
            "key2*"
          ]
        }
      }
    }
  ]
}
```

Le tableau suivant montre comment cette politique est AWS évaluée en fonction des valeurs des clés de condition figurant dans votre demande. Pour une instruction Refuser, Correspondance est refusée et Aucune correspondance n'est pas refusée, elle peut donc être autorisée par une autre instruction.

Condition de politique	Contexte de la demande	Résultat
`"ForAllValues:StringNotLike": { "aws:TagKeys": [ "key1*", "key2*" ] }`	`aws:TagKeys: - key1:legal`	Aucune correspondance Peut être autorisé par une autre instruction.
`"ForAllValues:StringNotLike": { "aws:TagKeys": [ "key1*", "key2*" ] }`	`aws:TagKeys: - key1:hr - key1:personnel`	Aucune correspondance Peut être autorisé par une autre instruction.

Condition de politique	Contexte de la demande	Résultat
`"ForAllValues:StringNotLike": { "aws:TagKeys": [ "key1*", "key2*" ] }`	`aws:TagKeys: - key1:hr - key2:audit`	Aucune correspondance Peut être autorisé par une autre instruction.
`"ForAllValues:StringNotLike": { "aws:TagKeys": [ "key1*", "key2*" ] }`	`aws:TagKeys: - key3:legal`	Match
`"ForAllValues:StringNotLike": { "aws:TagKeys": [ "key1*", "key2*" ] }`	Aucun `aws:TagKeys` dans le contexte de la demande.	Match

Exemple : politique de refus avec opérateur de jeu de conditions ForAnyValue

L'exemple de politique basée sur l'identité suivant interdit la création d'instantanés de volumes d'instance EC2 si des instantanés sont balisés à l'aide de l'une des clés de balise spécifiées dans la politique, `environment` ou `webserver`. La politique inclut l'opérateur d'ensemble `ForAnyValue` avec la clé de contexte `aws:TagKeys`, car la clé de contexte de la demande peut inclure plusieurs valeurs. Si votre demande de balisage inclut l'une des valeurs de clés de balise spécifiées dans la politique, la clé de contexte `aws:TagKeys` renvoie la valeur true en invoquant l'effet de la politique de refus.

> **Important**
>
> Cette politique ne permet aucune action. Utilisez cette stratégie conjointement à d'autres stratégies qui autorisent des actions spécifiques.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "ec2:CreateSnapshot",
        "ec2:CreateSnapshots"
      ],
      "Resource": "arn:aws:ec2:us-west-2::snapshot/*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:TagKeys": "webserver"
        }
      }
    }
  ]
}
```

Le tableau suivant montre comment cette politique est AWS évaluée en fonction des valeurs des clés de condition figurant dans votre demande. Pour une instruction Refuser, Correspondance est refusée et Aucune correspondance n'est pas refusée, elle peut donc être autorisée par une autre instruction.

Condition de politique	Contexte de la demande	Résultat
`"ForAnyValue:StringEquals": { "aws:TagKeys": "webserver" }`	`aws:TagKeys: - webserver`	Match

Condition de politique	Contexte de la demande	Résultat
`"ForAnyValue:StringEquals": { "aws:TagKeys": "webserver" }`	`aws:TagKeys: - environment - webserver - test`	Match
`"ForAnyValue:StringEquals": { "aws:TagKeys": "webserver" }`	`aws:TagKeys: - environment - test`	Aucune correspondance Peut être autorisé par une autre instruction.
`"ForAnyValue:StringEquals": { "aws:TagKeys": "webserver" }`	Aucun `aws:TagKeys` dans le contexte de la demande.	Aucune correspondance Peut être autorisé par une autre instruction.

Single-valued exemples de politiques clés contextuelles

L'ensemble suivant d'exemples de politiques montre comment créer des conditions de politique à l'aide de clés de contexte à valeur unique.

Exemple : plusieurs blocs de conditions avec des clés de contexte à valeur unique

Lorsqu'un bloc de conditions comporte plusieurs conditions, chacune associée à une seule clé de contexte, toutes les clés de contexte doivent aboutir à la valeur true pour que l'effet `Allow` ou `Deny` souhaité soit invoqué. Lorsque vous utilisez des opérateurs de condition de correspondance négative, la logique d'évaluation de la valeur de la condition est inversée.

L'exemple suivant permet aux utilisateurs de créer des volumes EC2 et d'appliquer des balises à des volumes pendant la création de volume. Le contexte de la demande doit inclure une valeur pour la clé de contexte `aws:RequestTag/project` et la valeur de la clé de contexte `aws:ResourceTag/environment` peut être n'importe quoi, sauf la production.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:CreateVolume",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:us-east-1:123456789012:volume/*",
      "Condition": {
        "StringLike": {
          "aws:RequestTag/project": "*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:us-east-1:123456789012:*/*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceTag/environment": "production"
        }
      }
    }
  ]
}
```

Le contexte de demande doit inclure une valeur de balise de projet et ne peut pas être créé pour qu'une ressource de production puisse invoquer l'effet `Allow`. Le volume EC2 suivant a été créé avec succès, car le nom du projet est `Feature3` avec une balise de ressource `QA`.

```
aws ec2 create-volume \
    --availability-zone us-east-1a \
    --volume-type gp2 \
    --size 80 \
```

```
    --tag-specifications 'ResourceType=volume,Tags=[{Key=project,Value=Feature3},
{Key=environment,Value=QA}]'
```

Exemple : un bloc de conditions avec plusieurs clés et valeurs de contexte à valeur unique

Lorsqu'un bloc de conditions contient plusieurs clés de contexte et que chaque clé de contexte possède plusieurs valeurs, chaque clé de contexte doit aboutir à true pour au moins une valeur de clé afin que l'effet `Allow` ou `Deny` souhaité soit invoqué. Lorsque vous utilisez des opérateurs de condition de correspondance négative, la logique d'évaluation de la valeur de clé de contexte est inversée.

L'exemple suivant permet aux utilisateurs de démarrer et d'exécuter des tâches sur des clusters Amazon Elastic Container Service.

- Le contexte de la demande doit inclure `production` OR `prod-backup` pour la clé de contexte `aws:RequestTag/environment`AND.
- La clé de contexte `ecs:cluster` garantit que les tâches sont exécutées sur les clusters ECS d'ARN `default1` OR `default2`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:RunTask",
        "ecs:StartTask"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/environment": [
            "production",
            "prod-backup"
          ]
        },
```

```
        "ArnEquals": {
          "ecs:cluster": [
            "arn:aws:ecs:us-east-1:111122223333:cluster/default1",
            "arn:aws:ecs:us-east-1:111122223333:cluster/default2"
          ]
        }
      }
    }
  ]
}
```

Le tableau suivant montre comment AWS évaluer cette politique en fonction des valeurs de clé de condition de votre demande.

Condition de politique	Contexte de la requête	Résultat
`"StringEquals": { "aws:RequestTag/environment": [ "production", "prod-backup" ] }, "ArnEquals": { "ecs:cluster": [ "arn:aws:ecs:us-east-1:111122223333:cluster/default1", "arn:aws:ecs:us-east-1:111122223333:cluster/default2" ] }`	`aws:RequestTag: environment:production ecs:cluster: arn:aws:ecs:us-east-1:111122223333:cluster/default1`	Correspondance
`"StringEquals": { "aws:RequestTag/environment": [ "production", "prod-backup" ]`	`aws:RequestTag: environment:prod-backup ecs:cluster:`	Correspondance

Condition de politique	Contexte de la requête	Résultat
}, "ArnEquals": { "ecs:cluster": ["arn:aws:ecs:us-east-1:111122223333:cluster/default1", "arn:aws:ecs:us-east-1:111122223333:cluster/default2"] }	arn:aws:ecs:us-east-1:111122223333:cluster/default2	
"StringEquals": { "aws:RequestTag/environment": ["production", "prod-backup"] }, "ArnEquals": { "ecs:cluster": ["arn:aws:ecs:us-east-1:111122223333:cluster/default1", "arn:aws:ecs:us-east-1:111122223333:cluster/default2"] }	aws:RequestTag: webserver:production ecs:cluster: arn:aws:ecs:us-east-1:111122223333:cluster/default2	Aucune correspondance

Condition de politique	Contexte de la requête	Résultat
`"StringEquals": { "aws:RequestTag/environment": [ "production", "prod-backup" ] }, "ArnEquals": { "ecs:cluster": [ "arn:aws:ecs:us-east-1:111122223333:cluster/default1", "arn:aws:ecs:us-east-1:111122223333:cluster/default2" ] }`	Aucun `aws:RequestTag` dans le contexte de la demande. `ecs:cluster arn:aws:ecs:us-east-1:111122223333:cluster/default2`	Aucune correspondance

Éléments des politiques IAM : variables et balises

Utilisez des variables de stratégie Gestion des identités et des accès AWS (IAM) comme espaces réservés lorsque vous ne connaissez pas la valeur exacte d'une ressource ou d'une clé de condition lorsque vous rédigez la politique.

> **Note**
>
> Si vous AWS ne parvenez pas à résoudre une variable, cela peut rendre l'instruction entière invalide. Par exemple, si vous utilisez la variable `aws:TokenIssueTime`, elle est résolue en une valeur uniquement lorsque le demandeur s'est authentifié à l'aide des informations d'identification temporaires (un rôle IAM). Pour éviter que les variables ne génèrent des instructions invalides, utilisez le... IfExists opérateur conditionnel.

Rubriques

- Introduction
- Utilisation de variables dans les politiques

- Les balises comme variables de la politique
- Éléments dans lesquels vous pouvez utiliser des variables de politique
- Variables de politique sans valeur
- Informations de demande que vous pouvez utiliser dans les variables de politique
- Spécification des valeurs par défaut
- Pour plus d'informations

Introduction

Dans les politiques IAM, de nombreuses actions vous permettent de fournir un nom pour des ressources spécifiques auxquelles vous voulez contrôler l'accès. Par exemple, la politique suivante permet aux utilisateurs d'afficher, de lire et d'écrire les objets dans le compartiment S3 `amzn-s3-demo-bucket` pour les projets `marketing`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"],
      "Condition": {"StringLike": {"s3:prefix": ["marketing/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/marketing/*"]
    }
  ]
}
```

Dans certains cas, vous ne connaissez pas le nom exact de la ressource lorsque vous écrivez la politique. Vous souhaiterez peut-être généraliser la politique de façon à ce qu'elle fonctionne pour

de nombreux utilisateurs sans avoir à faire une copie unique de celle-ci pour chaque utilisateur. Au lieu de créer une politique distincte pour chaque utilisateur, nous vous recommandons de créer une politique de groupe unique qui s'applique à tous les utilisateurs du groupe.

Utilisation de variables dans les politiques

Vous pouvez définir des valeurs dynamiques dans des politiques en utilisant des variables de politique qui définissent des espaces réservés dans une politique.

Les variables sont marquées à l'aide d'un préfixe **$** suivi d'une paire d'accolades (**{ }**) qui incluent le nom de variable de la valeur issue de la demande.

Lorsque la politique est évaluée, les variables de politique sont remplacées par les valeurs provenant des clés contextuelles conditionnelles passées dans la requête. Les variables peuvent être utilisées dans les politiques basées sur l'identité, les politiques de ressources, les politiques de contrôle des services, les politiques de session et les politiques de terminaux VPC. Identity-based les politiques utilisées comme limites d'autorisations prennent également en charge les variables de stratégie.

Les clés contextuelles de condition globale peuvent être utilisées comme variables dans les demandes relatives à l'ensemble AWS des services. Les clés de condition spécifiques à un service peuvent également être utilisées comme variables lors de l'interaction avec des ressources AWS , mais elles ne sont disponibles que lorsque des requêtes sont effectuées auprès de ressources qui les prennent en charge. Pour obtenir la liste des clés contextuelles disponibles pour chaque AWS service et ressource, consultez la référence d'autorisation des services . Dans certains cas, vous ne pouvez pas renseigner les clés contextuelles des conditions globales avec une valeur. Pour en savoir plus sur chaque clé, consultez Clés de contexte de condition globale AWS.

> **Important**
>
> - Les noms des clés ne sont pas sensibles à la casse. Par exemple, `aws:CurrentTime` équivaut à `AWS:currenttime`.
> - Vous pouvez utiliser n'importe quelle clé de condition à valeur unique comme variable. Vous ne pouvez pas utiliser de clé de condition à valeurs multiples en tant que variable.

L'exemple suivant illustre une politique pour un rôle ou un utilisateur IAM qui remplace un nom de ressource spécifique par une variable de politique. Vous pouvez réutiliser cette politique en tirant parti de la clé de condition `aws:PrincipalTag`. Lorsque cette politique est évaluée,

`${aws:PrincipalTag/team}` autorise les actions uniquement si le nom du compartiment se termine par un nom d'équipe issu de la balise de principal `team`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"],
      "Condition": {"StringLike": {"s3:prefix": ["${aws:PrincipalTag/team}/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/${aws:PrincipalTag/team}/*"]
    }
  ]
}
```

La variable est marquée à l'aide d'un préfixe $ suivi d'une paire d'accolades ({ }). À l'intérieur des caractères ${ }, vous pouvez inclure le nom de la valeur de la demande que vous souhaitez utiliser dans la politique. Les valeurs disponibles sont détaillées plus loin sur cette page.

Pour obtenir la liste détaillé de cette clé de condition globale, consultez les lois :PrincipalTag/clé-balise dans la liste des clés de condition globales.

> **Note**
>
> Pour utiliser des variables de politique, vous devez inclure l'élément `Version` dans une instruction et la version doit être définie sur une version prenant en charge de telles variables. Les variables ont été introduites dans la version `2012-10-17`. Les versions antérieures du langage de politique ne prennent pas en charge les variables de politique. Si vous n'incluez pas l'élément `Version` et que la valeur correspond à une date de version appropriée, les

variables telles que `${aws:username}` sont traitées comme des chaînes littérales dans la politique.
Un élément de politique `Version` varie d'une version de politique. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. En revanche, une version de politique est créée lorsque vous apportez des modifications à une politique gérée par le client dans IAM. La politique modifiée ne remplace pas la politique existante. À la place, IAM crée une nouvelle version de la politique gérée. Pour en savoir plus sur l'élément de politique `Version`, consultez the section called "Version". Pour en savoir plus sur les versions de politiques, consultez the section called "Gestion des versions des politiques IAM".

Une politique qui permet à un principal d'obtenir des objets à partir du chemin /David d'un compartiment S3 ressemble à ceci :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/David/*"
      ]
    }
  ]
}
```

Si cette politique est attaché à l'utilisateur `David`, celui-ci obtient les objets de son propre compartiment S3, mais il vous faudra créer une politique distincte pour chaque utilisateur incluant le nom utilisateur. Vous devez ensuite attacher chaque politique aux utilisateurs individuels.

À l'aide d'une variable de politique, vous pouvez créer des politiques réutilisables. La politique suivante permet à un utilisateur d'obtenir des objets à partir d'un compartiment Amazon S3 si la valeur de la clé de balise `aws:PrincipalTag` correspond à la valeur de la clé de balise `owner` transmise dans la demande.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Sid": "AllowUnlessOwnedBySomeoneElse",
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["*"],
    "Condition": {
        "StringEquals": {
          "s3:ExistingObjectTag/owner": "${aws:PrincipalTag/owner}"
        }
      }
    }
  ]
}
```

Lorsque vous utilisez une variable de politique à la place d'un utilisateur, il n'est pas nécessaire d'avoir une politique distincte pour chaque utilisateur. Dans l'exemple suivant, la politique est attachée à un rôle IAM endossé par les Product Managers à l'aide d'informations d'identification de sécurité temporaires. Lorsqu'un utilisateur demande à ajouter un objet Amazon S3, IAM remplace la valeur de balise `dept` de la demande actuelle pour la variable `${aws:PrincipalTag}`, puis évalue la politique.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOnlyDeptS3Prefix",
            "Effect": "Allow",
            "Action": [
```

```
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/${aws:PrincipalTag/dept}/*"
            ]
        }
    ]
}
```

Les balises comme variables de la politique

Dans certains AWS services, vous pouvez associer vos propres attributs personnalisés aux ressources créées par ces services. Par exemple, vous pouvez appliquer des balises aux compartiments Amazon S3 ou aux utilisateurs IAM. Ces balises sont des paires clé-valeur. Vous définissez le nom de clé de balise et la valeur qui est associée à ce nom de clé. Par exemple, vous pouvez créer une balise avec une clé **`department`** et une valeur **`Human Resources`**. Pour plus d'informations sur le balisage des entités IAM, consultez Tags pour Gestion des identités et des accès AWS resources. Pour plus d'informations sur le balisage des ressources créées par d'autres services AWS , reportez-vous à la documentation de ce service. Pour plus d'informations sur l'utilisation de Tag Editor, consultez Utilisation de Tag Editor dans le Guide de l'utilisateur Console de gestion AWS .

Vous pouvez baliser des ressources IAM pour simplifier la découverte, l'organisation et le suivi de vos ressources IAM. Vous pouvez aussi baliser les identités IAM pour contrôler l'accès aux ressources ou au balisage lui-même. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Éléments dans lesquels vous pouvez utiliser des variables de politique

Vous pouvez utiliser des variables de politique dans l'élément `Resource` et les comparaisons de chaîne de l'élément `Condition`.

Élément de ressource

Vous pouvez utiliser une variable de politique dans l'élément `Resource`, mais uniquement dans la partie ressource de l'ARN. Cette partie de l'ARN apparaît après le cinquième deux-points (:). Vous ne pouvez pas utiliser une variable pour remplacer des parties de l'ARN avant le cinquième deux-points, par exemple le service ou le compte. Pour de plus amples informations sur le format ARN, veuillez consulter ARN IAM.

Pour remplacer une partie d'un ARN avec une valeur de balise, encadrez le préfixe et le nom de clé avec `${ }`. Par exemple, l'élément de ressource suivant fait référence à un seul compartiment qui est nommé à l'identique de la valeur de la balise du service de l'utilisateur demandeur.

```
"Resource": ["arn:aws::s3:::amzn-s3-demo-bucket/
${aws:PrincipalTag/department}"]
```

De nombreuses AWS ressources utilisent des ARN contenant un nom créé par l'utilisateur. La politique IAM suivante garantit que seuls les utilisateurs concernés dont les valeurs de balises access-project, access-application et access-environment sont correspondantes peuvent modifier leurs ressources. En outre, en utilisant le caractère générique *, ils peuvent autoriser des suffixes de nom de ressources personnalisés.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessBasedOnArnMatching",
      "Effect": "Allow",
      "Action": [
        "sns:CreateTopic",
        "sns:DeleteTopic"],
      "Resource": ["arn:aws:sns:*:*:${aws:PrincipalTag/access-project}-
${aws:PrincipalTag/access-application}-${aws:PrincipalTag/access-environment}-*"
      ]
    }
  ]
}
```

Élément de condition

Vous pouvez utiliser une variable de politique pour les valeurs `Condition` dans n'importe quelle condition impliquant les opérateurs de chaîne ou les opérateurs ARN. Les opérateurs de chaîne incluent `StringEquals`, `StringLike` et `StringNotLike`. Les opérateurs ARN incluent `ArnEquals` et `ArnLike`. Vous ne pouvez pas utiliser une variable de politique avec d'autres opérateurs, tels que les opérateurs `Numeric`, `Date`, `Boolean`, `Binary`, `IP Address` ou `Null`.

Pour de plus amples informations sur les opérateurs de condition, veuillez consulter Éléments de politique JSON IAM : Opérateurs de condition.

Lorsque vous faites référence à une balise dans une expression de l'élément `Condition`, utilisez le préfixe et le nom de clé pertinents comme clé de condition. Utilisez ensuite la valeur que vous souhaitez tester dans la valeur de condition.

Par exemple, l'exemple de politique suivant autorise l'accès complet aux utilisateurs, mais uniquement si la balise `costCenter` est attachée à l'utilisateur. La balise doit également avoir la valeur `12345` ou `67890`. Si la balise n'a pas de valeur, ou toute autre valeur, la demande échoue.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
          "iam:*user*"
       ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:ResourceTag/costCenter": [ "12345", "67890" ]
        }
      }
    }
  ]
}
```

Variables de politique sans valeur

Lorsque les variables de politique font référence à une clé contextuelle de condition qui n'a aucune valeur ou qui n'est pas présente dans le contexte d'autorisation d'une requête, la valeur est effectivement nulle. Il n'existe pas de valeur égale ou similaire. Les clés contextuelles de condition peuvent ne pas être présentes dans le contexte d'autorisation lorsque :

- Vous utilisez des clés contextuelles de condition spécifiques au service dans les requêtes adressées à des ressources qui ne prennent pas en charge cette clé de condition.

- Les balises sur les principaux, les sessions, les ressources ou les demandes IAM ne sont pas présentes.
- Les autres cas répertoriés pour chaque condition globale sont incluses dans le contexte AWS clés de contexte de condition globale.

Lorsque vous utilisez une variable sans valeur dans l'élément de condition d'une politique IAM, Éléments de politique JSON IAM : Opérateurs de condition comme `StringEquals` ou `StringLike` ne correspondent pas et la déclaration de politique ne prend pas effet.

Les opérateurs de condition inversés comme `StringNotEquals` ou `StringNotLike` correspondent à une valeur nulle, car la valeur de la clé conditionnelle par rapport à laquelle ils testent n'est pas égale ou similaire à la valeur effectivement nulle.

Dans l'exemple suivant, `aws:principaltag/Team` doit être égal à `s3:ExistingObjectTag/Team` pour autoriser l'accès. L'accès est explicitement refusé lorsque `aws:principaltag/Team` n'est pas défini. Si une variable qui n'a aucune valeur dans le contexte d'autorisation est utilisée dans le cadre de l'élément `Resource` or `NotResource` d'une politique, la ressource qui inclut une variable de politique sans valeur ne correspondra à aucune ressource.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
   {
    "Effect": "Deny",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
      "StringNotEquals": {
        "s3:ExistingObjectTag/Team": "${aws:PrincipalTag/Team}"
       }
      }
    }
  ]
}
```

Informations de demande que vous pouvez utiliser dans les variables de politique

Vous pouvez utiliser l'élément `Condition` d'une politique JSON pour comparer des clés dans le contexte de demande avec les valeurs de clé spécifiées dans votre politique. Lorsque vous utilisez une variable de politique, elle AWS remplace la variable dans votre politique par une valeur issue de la clé de contexte de la demande.

Valeurs de la clé du principal

Les valeurs de `aws:username`, `aws:userid` et `aws:PrincipalType` dépendent du type de principal ayant initialisé la demande. Par exemple, la demande peut être effectuée à l'aide des informations d'identification d'un utilisateur IAM, d'un rôle IAM ou de l' Utilisateur racine d'un compte AWS. La liste de suivante affiche les valeurs de ces clés pour différents types de principaux.

- Utilisateur racine d'un compte AWS
  - `aws:username` : (absent)
  - `aws:userid`: Compte AWS Identifiant
  - `aws:PrincipalType`: `Account`
- Utilisateur IAM
  - `aws:username`: *`IAM-user-name`*
  - `aws:userid`: ID unique
  - `aws:PrincipalType`: `User`
- AWS STS utilisateur principal fédéré
  - `aws:username` : (absent)
  - `aws:userid`: *`account`*:*`caller-specified-name`*
  - `aws:PrincipalType`: `FederatedUser`
- Principaux fédérés OIDC et SAML

> **Note**
>
> Pour plus d'informations sur les clés de politique disponibles lors de l'utilisation de la fédération OIDC, consultez Fédération OIDC.

  - `aws:username` : (absent)
  - `aws:userid` : (absent)

    - `aws:PrincipalType`: `AssumedRole`
- Rôle endossé
    - `aws:username` : (absent)
    - `aws:userid`: *`role-id`*:*`caller-specified-role-name`*
    - `aws:PrincipalType`: `Assumed role`
- Rôle affecté à une instance Amazon EC2
    - `aws:username` : (absent)
    - `aws:userid`: *`role-id`*:*`ec2-instance-id`*
    - `aws:PrincipalType`: `Assumed role`
- Appelant anonyme (Amazon SQS, Amazon SNS et Amazon S3 uniquement)
    - `aws:username` : (absent)
    - `aws:userid` : (absent)
    - `aws:PrincipalType`: `Anonymous`

Pour les éléments de cette liste, notez les points suivants :

- absent signifie que la valeur ne figure pas dans les informations de la demande en cours et par conséquent, toute tentative de correspondance échoue et rend l'instruction non valide.
- *`role-id`*est un identifiant unique attribué à chaque rôle lors de sa création. Vous pouvez afficher l'ID du rôle à l'aide de la AWS CLI commande suivante : `aws iam get-role --role-name` *`rolename`*
- *`caller-specified-name`*et *`caller-specified-role-name`* sont des noms transmis par le processus appelant (comme une application ou un service) lorsqu'il passe un appel pour obtenir des informations d'identification temporaires.
- *`ec2-instance-id`*est une valeur attribuée à l'instance lors de son lancement et apparaît sur la page Instances de la console Amazon EC2. Vous pouvez également afficher l'ID de l'instance en exécutant la AWS CLI commande suivante : `aws ec2 describe-instances`

Informations relatives aux principaux fédérés disponibles dans les demandes

Les principaux fédérés sont des utilisateurs authentifiés à l'aide d'un système autre qu'IAM. Par exemple, une entreprise peut disposer d'une application à utiliser en interne qui passe des appels à AWS. L'attribution d'une identité IAM à chaque utilisateur de l'entreprise qui utilise l'application

peut ne pas convenir. Au lieu de cela, l'entreprise peut utiliser une application proxy (intermédiaire) qui dispose d'une identité IAM unique ou un fournisseur d'identité (IdP) SAML. L'application proxy ou l'IdP SAML authentifie les utilisateurs individuels via le réseau d'entreprise. Une application proxy peut ensuite utiliser son identité IAM pour obtenir des informations d'identification de sécurité temporaires pour des utilisateurs individuels. Un IdP SAML peut en effet échanger des informations d'identité contre des informations d'identification de sécurité AWS temporaires. Les informations d'identification temporaires peuvent ensuite être utilisées pour accéder aux AWS ressources.

De même, vous pouvez créer une application pour appareil mobile qui doit accéder aux ressources AWS . Dans ce cas, vous pouvez utiliser la fédération OIDC dans laquelle l’application authentifie l’utilisateur à l’aide d’un fournisseur d’identité connu tel que Login with Amazon, Amazon Cognito, Facebook ou Google. L'application peut ensuite utiliser les informations d'authentification de l'utilisateur à partir de ces fournisseurs pour obtenir les informations d'identification de sécurité temporaires permettant d'accéder aux ressources AWS .

La méthode recommandée pour utiliser la fédération OIDC consiste à tirer parti d'Amazon Cognito et des SDK AWS mobiles. Pour plus d’informations, consultez les ressources suivantes :

- Guide de l'utilisateur d'Amazon Cognito
- Scénarios courants d'informations d'identification temporaires

Caractères spéciaux

Quelques variables de politique prédéfinies spéciales sont dotées de valeurs fixes qui vous permettent de représenter des caractères qui, sinon, ont une signification spéciale. Si ces caractères spéciaux font partie de la chaîne que vous essayez de faire correspondre et qu'ils sont insérés littéralement, ils sont susceptibles d'être mal interprétés. Par exemple, un astérisque (*) inséré dans la chaîne sera interprété comme un caractère générique, ce qui correspond à n'importe quel caractère, au lieu d'un caractère * littéral. Dans ces cas, vous pouvez utiliser les variables de politique prédéfinies suivantes :

- ${*} – permet d'obtenir le caractère * (astérisque).
- ${?} – permet d'obtenir le caractère ? (point d'interrogation).
- ${$} – permet d'obtenir le caractère $ (dollar).

Ces variables de politique prédéfinies peuvent être utilisées dans n'importe quelle chaîne acceptant les variables de politique standard.

Spécification des valeurs par défaut

Pour ajouter une valeur par défaut à une variable, entourez la valeur par défaut de guillemets simples (`' '`), puis séparez le texte de la variable et la valeur par défaut par une virgule et une espace (`, `).

Par exemple, si un principal est labelisé avec l'interface `team=yellow`, il peut accéder au compartiment Amazon S3 `ExampleCorp's` nommé `amzn-s3-demo-bucket-yellow`. Une politique avec cette ressource permet aux membres de l'équipe d'accéder à leur compartiment d'équipe, mais pas à ceux des autres équipes. Pour les utilisateurs sans identifications d'équipe, il définit une valeur par défaut de l'interface `company-wide` pour le nom du compartiment. Ces utilisateurs peuvent accéder uniquement au compartiment `amzn-s3-demo-bucket-company-wide` où ils peuvent afficher des informations générales, telles que des instructions pour rejoindre une équipe.

```
"Resource":"arn:aws:s3:::amzn-s3-demo-bucket-${aws:PrincipalTag/team, 'company-wide'}"
```

Pour plus d'informations

Pour plus d'informations sur les politiques, consultez les ressources suivantes :

- Politiques et autorisations dans Gestion des identités et des accès AWS
- Exemples de politiques basées sur l'identité IAM
- Référence de l'élément de politique JSON IAM
- Logique d'évaluation de politiques
- Fédération OIDC

Éléments de politique JSON IAM : Types de données pris en charge

Cette section répertorie les types de données pris en charge lors de la spécification de valeurs dans des politiques JSON. Le langage de politique ne prend pas en charge tous les types d'éléments de politique ; reportez-vous aux sections précédentes pour plus d'informations sur chaque élément.

- Chaînes
- Nombres (Ints et Floats)
- Booléen
- Null
- Lists

- Mappages
- Structs (simples mappages intégrés)

Le tableau suivant mappe chaque type de données à la sérialisation. Notez que toutes les politiques doivent être intégrées UTF-8. Pour plus d'informations sur les types de données JSON, consultez la RFC 4627.

Type	JSON
String	String
Entier	Number
Float	Number
Booléen	true false
Null	null
Date	Chaîne conforme au profil W3C d'ISO 8601
IpAddress	Chaîne conforme à RFC 4632
List	Tableau
Objet	Objet

Logique d'évaluation de politiques

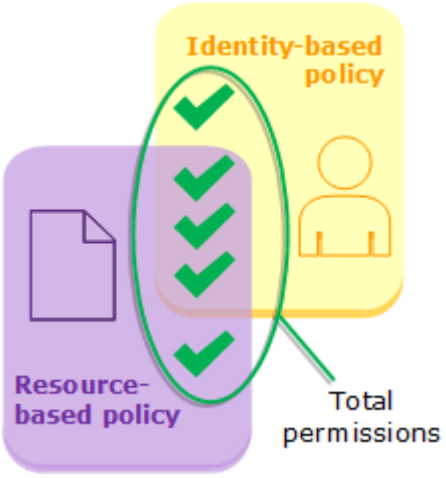
Lorsqu'un principal essaie d'utiliser l' Console de gestion AWS AWS API, ou le AWS CLI, ce principal envoie une demande à AWS. Lorsqu'un AWS service reçoit la demande, AWS effectue plusieurs étapes pour déterminer s'il convient d'autoriser ou de refuser la demande.

1. Authentification : authentifie d' AWS abord le principal qui fait la demande, si nécessaire. Cette étape n'est pas nécessaire pour quelques services, tels qu'Amazon S3, qui autorisent certaines demandes provenant d'utilisateurs anonymes.
2. Traitement du contexte de la demande— AWS traite les informations recueillies dans la demande afin de déterminer quelles politiques s'appliquent à la demande.

3. Comment ? AWS la logique du code d'application évalue les demandes pour autoriser ou refuser l'accès— AWS évalue tous les types de politiques et l'ordre des politiques influe sur la façon dont elles sont évaluées. AWS traite ensuite les politiques en fonction du contexte de la demande pour déterminer si la demande est autorisée ou refusée.

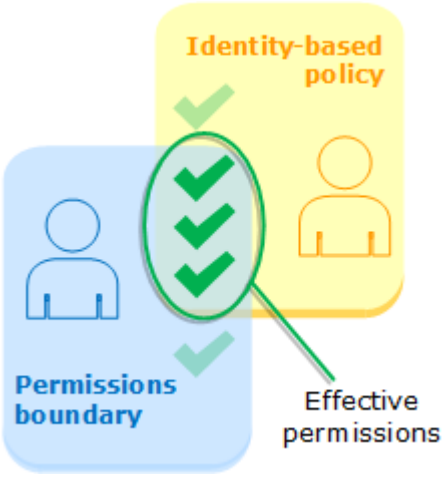
Évaluation des politiques basées sur l'identité avec des politiques basées sur des ressources

Identity-based les politiques et les politiques basées sur les ressources accordent des autorisations aux identités ou aux ressources auxquelles elles sont associées. Lorsqu'une entité IAM (utilisateur ou rôle) demande l'accès à une ressource au sein du même compte, AWS elle évalue toutes les autorisations accordées par les politiques basées sur l'identité et les ressources. Les autorisations résultantes sont l'union des autorisations des deux types. Si une action est autorisée par une stratégie basée sur l'identité, une politique basée sur les ressources, ou les deux, l'action est autorisée. AWS Un refus explicite dans l'une ou l'autre de ces stratégies remplace l'autorisation.



Évaluation des politiques basées sur l'identité avec des limites d'autorisations

Lors de l' AWS évaluation des politiques basées sur l'identité et des limites d'autorisations pour un utilisateur, les autorisations qui en résultent sont à l'intersection des deux catégories. En d'autres termes, lorsque vous ajoutez une limite d'autorisations à un utilisateur avec les politiques basées sur l'identité existantes, vous pouvez réduire les actions exécutées par l'utilisateur. Sinon, lorsque vous supprimez une limite d'autorisations à partir d'un utilisateur, vous pouvez augmenter les actions qu'il peut exécuter. Un refus explicite dans l'une ou l'autre de ces stratégies remplace l'autorisation. Pour afficher des informations sur la façon dont les autres types de politique sont évalués avec des limites d'autorisations, consultez Évaluation des autorisations effectives avec limites.



Évaluer les politiques basées sur l'identité avec AWS Organizations SCP ou RCP

Lorsqu'un utilisateur appartient à un compte membre d'une organisation et accède à une ressource pour laquelle aucune politique basée sur les ressources n'a été configurée, les autorisations résultantes sont l'intersection des politiques de l'utilisateur, des politiques de contrôle des services (SCP) et de la politique de contrôle des ressources (RCP). Cela signifie qu'une action doit être autorisée par les trois types de politique. Un refus explicite dans la politique basée sur l'identité, une SCP ou une RCP annule l'autorisation.



Vous pouvez savoir si votre compte est membre d'une organisation dans AWS Organizations. Les membres de l'organisation pourraient être affectés par une SCP ou une RCP. Pour afficher ces données à l'aide de la AWS CLI commande ou de l'opération AWS API, vous devez disposer des autorisations nécessaires à l'`organizations:DescribeOrganization`action pour votre AWS Organizations entité. Vous devez disposer d'autorisations supplémentaires pour effectuer l'opération dans la AWS Organizations console. Pour savoir si un SCP ou un RCP refuse

l'accès à une demande spécifique, ou pour modifier vos autorisations effectives, contactez votre AWS Organizations administrateur.

Traitement du contexte de la demande

Lorsqu'il AWS évalue et autorise une demande, il assemble les informations de la demande dans un contexte de demande. Le contexte de la demande contient toutes les informations pouvant être utilisées dans l’évaluation de la politique.

- Principal : l’utilisateur, le rôle ou le principal utilisateur fédéré à l’origine de la demande. Les informations sur le principal incluent les politiques associées à ce dernier.
- Actions : une ou plusieurs actions que le principal souhaite exécuter.
- Ressources : un ou plusieurs objets de AWS ressources sur lesquels les actions ou opérations sont effectuées.
- Données de ressources : données liées à la ressource qui est demandée. Par exemple, ces informations peuvent inclure le nom d'une table DynamoDB ou une balise sur une instance Amazon EC2.
- Données d'environnement : informations sur l'adresse IP, l'agent utilisateur, le statut SSL ou le moment de la journée.

Ces informations sont comparées aux politiques applicables afin de déterminer s’il convient d’accepter ou de refuser la demande. Vous pouvez organiser ces informations de propriété à l'aide du modèle PARC (Principal, Action, Resource, and Condition) afin de mieux comprendre comment AWS les politiques sont évaluées.

Comprendre le modèle PARC

Le modèle PARC représente le contexte de la demande sur la base des quatre éléments JSON du langage de politique :

- Principal : l’entité à l’origine de la demande. Un principal représente un utilisateur humain ou une charge de travail programmatique qui peut être authentifié et autorisé à effectuer des actions dans Comptes AWS.
- Action : l’opération en cours d’exécution. Souvent, l’action sera mappée à une action API.
- Resource ; la ressource AWS sur laquelle l’action est effectuée.
- Condition : contraintes supplémentaires qui doivent être respectées pour que la demande soit acceptée.

Voici un exemple illustrant comment le modèle PARC pourrait représenter un contexte de demande :

```
Principal: AIDA123456789EXAMPLE
Action: s3:CreateBucket
Resource: arn:aws:s3:::amzn-s3-demo-bucket1
Context:
- aws:UserId=AIDA123456789EXAMPLE:BobsSession
- aws:PrincipalAccount=123456789012
- aws:PrincipalOrgId=o-example
- aws:PrincipalARN=arn:aws:iam::AIDA123456789EXAMPLE:role/HR
- aws:MultiFactorAuthPresent=true
- aws:CurrentTime=...
- aws:EpochTime=...
- aws:SourceIp=...
- aws:PrincipalTag/dept=123
- aws:PrincipalTag/project=blue
- aws:RequestTag/dept=123
```

Importance du contexte de la demande

Il est essentiel de comprendre le contexte de la demande et son interaction avec l’évaluation des politiques pour :

- Résoudre les problèmes d’accès
- Concevoir des politiques efficaces et sécurisées
- Comprendre l’étendue complète des autorisations accordées par une politique
- Prédire le résultat des évaluations des politiques dans différents scénarios

En visualisant le contexte de la demande à l'aide du modèle PARC, vous pouvez plus facilement comprendre comment sont prises les AWS décisions d'autorisation et concevoir vos politiques de manière plus efficace.

Comment AWS utilise le contexte de la demande

Lors de l'évaluation des politiques, AWS compare les informations contenues dans le contexte de la demande avec les informations spécifiées dans toutes les politiques applicables. Cela inclut les politiques basées sur l'identité, les politiques basées sur les ressources, les limites des autorisations IAM, les Organizations, les Organizations et les politiques de SCPs session. RCPs

Pour chaque type de politique, AWS utilise le contexte de la demande pour vérifier :

- Si la politique s’applique à la demande en fonction du principal.
- Si l’action demandée est autorisée sur la ressource spécifiée.
- Si les conditions spécifiées dans la politique sont remplies par le contexte de la demande.

La manière dont AWS les politiques sont évaluées dépend des types de politiques qui s'appliquent au contexte de la demande. Ces types de politique sont utilisables dans un seul Compte AWS. Pour plus d'informations sur ces types de politique, consultez Politiques et autorisations dans Gestion des identités et des accès AWS. Pour savoir comment AWS évalue les politiques d'accès entre comptes, voir. Cross-account logique d'évaluation des politiques

- AWS Organizations politiques de contrôle des ressources (RCPs) : AWS Organizations RCPs spécifiez les autorisations maximales disponibles pour les ressources au sein des comptes d'une organisation ou d'une unité organisationnelle (UO). RCPs s'appliquent aux ressources des comptes membres et ont un impact sur les autorisations effectives accordées aux directeurs Utilisateur racine d'un compte AWS, y compris, qu'ils appartiennent ou non à votre organisation. RCPs ne s'appliquent pas aux ressources du compte de gestion de l'organisation ni aux appels effectués par des rôles liés à un service. Si une RCP est présente, les autorisations accordées par les politiques basées sur les identités et les ressources aux ressources de vos comptes membres ne sont effectives que si la RCP autorise l’action.
- AWS Organizations politiques de contrôle des services (SCPs) : AWS Organizations SCPs spécifiez les autorisations maximales disponibles pour les principaux au sein des comptes d'une organisation ou d'une unité organisationnelle (UO). SCPs s'appliquent aux directeurs des comptes des membres, y compris à chacun Utilisateur racine d'un compte AWS d'entre eux. Si une SCP est présente, les autorisations accordées par les politiques basées sur les identités et les ressources aux principaux de vos comptes membres ne sont effectives que si la SCP autorise l’action. Les seules exceptions sont les principaux du compte de gestion de l’organisation et les rôles liés à un service.
- Politiques basées sur les ressources : les politiques basées sur les ressources accordent des autorisations aux principaux spécifiés dans la politique. Les autorisations définissent ce que le principal peut faire avec la ressource à laquelle la politique est attachée.
- Limites des autorisations : les limites d’autorisations sont une fonctionnalité qui définit les autorisations maximales qu’une politique basée sur les identités peut accorder à une entité IAM (utilisateur ou rôle). Lorsque vous définissez une limite d’autorisations pour une entité, l’entité peut effectuer uniquement les actions autorisées par ses deux ses politiques basées sur l’identité et ses limites d’autorisations. Dans certains cas, un refus implicite dans une limite d'autorisations

peut limiter les autorisations accordées par une politique basée sur les ressources. Pour de plus amples informations, veuillez consulter Comment ? AWS la logique du code d'application évalue les demandes pour autoriser ou refuser l'accès.

- Politiques basées sur l'identité : les politiques basées sur l'identité sont attachées à une identité IAM (utilisateur, groupe d'utilisateurs ou rôle) et octroient des autorisations aux entités IAM (utilisateurs et rôles). Si seules les politiques basées sur l'identité s'appliquent à une demande, alors AWS vérifie toutes ces politiques pour au moins une d'entre elles. `Allow`
- Politiques de session : les politiques de session sont des politiques que vous transmettez en tant que paramètres lorsque vous créez par programmation une session temporaire pour un rôle ou une session d'utilisateur fédéré. Pour créer une session de rôle par programmation, utilisez l'une des opérations d'API `AssumeRole*`. Lorsque vous procédez ainsi et transmettez de politiques de session, les autorisations de session obtenues sont une combinaison de la politique basée sur l'identité de l'entité IAM et des politiques de session. Pour créer une session d'utilisateur fédéré, vous utilisez des clés d'accès d'utilisateur IAM pour appeler par programmation l'opération d'API `GetFederationToken`. Pour de plus amples informations, veuillez consulter Politiques de session.

Pour rappel, un refus explicite dans l'une de ces politiques remplace l'autorisation.

> **Note**
>
> AWS Organizations les politiques déclaratives vous permettent de déclarer et d'appliquer de manière centralisée la configuration souhaitée pour une donnée Service AWS à grande échelle au sein d'une organisation. Étant donné que les politiques déclaratives sont appliquées directement au niveau du service, elles n'ont pas d'impact direct sur les demandes d'évaluation des politiques et ne sont pas incluses dans le contexte de la demande. Pour plus d'informations, consultez la section Politiques déclaratives dans le guide de l' AWS Organizations utilisateur.

Exemple d'évaluation des politiques à l'aide du modèle PARC

Pour illustrer comment le contexte de la demande interagit avec l'évaluation de la politique, prenons un exemple de politique :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:CreateBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/dept": "123"
        }
      }
    }
  ]
}
```

Dans cet exemple, la politique n'autoriserait l'action `CreateBucket` que si le contexte de la demande inclut une valeur `aws:PrincipalTag/dept` de « 123 » et si la ressource correspond au nom du compartiment `amzn-s3-demo-bucket1`. Le tableau suivant montre comment AWS utilise le contexte de la demande pour évaluer cette politique et prendre des décisions d'autorisation.

Politique	Contexte de la requête	Résultat de l'évaluation

`"StringEquals": { "aws:PrincipalTag/dept": "123" }`	`Principal: AIDA12345 6789EXAMPLE`
	`Action: s3:Create Bucket`
	`Resource: arn:aws:s 3:::amzn-s3-demo-b ucket1`
	`Context: - aws:PrincipalTag/ dept=123`
		Match

`"StringEquals": { "aws:PrincipalTag/dept": "123"`	`Principal: AIDA12345 6789EXAMPLE`	Aucune correspondance

Politique	Contexte de la requête	Résultat de l'évaluation

`}`	`Action: s3: DeleteBucket`
	`Resource: arn:aws:s3:::amzn-s3-demo-bucket1`
	`Context:`
	`- aws:PrincipalTag/dept=123`	

`"StringEquals": {`
`"aws:PrincipalTag/dept": "123"`
`}`
	`Principal: AIDA123456789EXAMPLE`
	`Action: s3:CreateBucket`
	`Resource: arn:aws:s3:::amzn-s3-demo-bucket1`
	`Context:`
	`- aws:PrincipalTag/dept= 321`
		Aucune correspondance

`"StringEquals": {`
`"aws:PrincipalTag/dept": "123"`
`}`
	`Principal: AIDA123456789EXAMPLE`
	`Action: s3:CreateBucket`
	`Resource: arn:aws:s3:::amzn-s3-demo-bucket1`
	`Context:`
	Aucun `aws:PrincipalTag` dans la demande.
		Aucune correspondance

Comment ? AWS la logique du code d'application évalue les demandes pour autoriser ou refuser l'accès

Le code AWS d'exécution décide si une demande envoyée à AWS doit être acceptée ou refusée. AWS évalue toutes les politiques applicables au contexte de la demande. Voici un résumé de la logique d'évaluation des AWS politiques.

- Par défaut, toutes les demandes sont implicitement rejetées à l'exception de l' Utilisateur racine d'un compte AWS, qui dispose d'un accès complet.
- Les demandes doivent être explicitement autorisées par une politique ou un ensemble de politiques suivant la logique d'évaluation ci-dessous pour être autorisées.
- Un refus explicite remplace une autorisation explicite.

L'évaluation de la politique peut varier selon que la demande concerne un seul compte ou plusieurs comptes. Pour plus d'informations sur la manière dont une décision d'évaluation de politique est prise pour un rôle IAM ou un utilisateur au sein d'un seul compte, consultez Évaluation des politiques pour les demandes au sein d'un même compte. Pour plus de détails sur la manière dont une décision d'évaluation des politiques est prise pour les demandes entre comptes, consultez Cross-account logique d'évaluation des politiques.

- Deny evaluation (Refuser l'évaluation) : par défaut, toutes les demandes sont refusées. Il s'agit d'un refus implicite. Le code AWS d'application évalue toutes les politiques du compte qui s'appliquent à la demande. Il s'agit notamment des AWS Organizations SCP et des RCP, des politiques basées sur les ressources, des politiques basées sur l'identité, des limites d'autorisations IAM et des politiques de session. Dans toutes ces politiques, le code d'application recherche une instruction `Deny` (Refuser) qui s'applique à la demande. Ceci est appelé un refus explicite. Si le code d'application trouve un refus explicite qui s'applique, il retourne la décision finale Refuser. S'il n'y a pas de refus explicite, l'évaluation du code d'application se poursuit.
- AWS Organizations RCP — Le code d'application évalue les politiques de contrôle AWS Organizations des ressources (RCP) qui s'appliquent à la demande. Les RPC s'appliquent aux ressources du compte auquel elles sont rattachées. Si le code d'application ne trouve aucune instruction `Allow` applicable dans les RPC, le code d'application retourne la décision finale Refuser. Notez qu'une politique gérée AWS appelée `RCPFullAWSAccess` est automatiquement créée et attachée à chaque entité de votre organisation, y compris la racine, chaque unité d'organisation et Compte AWS lorsque les RCP sont activées. `RCPFullAWSAccess` ne peut pas

être détachée, il y aura donc toujours une instruction `Allow`. S'il n'y a pas de RCP, ou si la RCP autorise l'action demandée, l'évaluation du code d'application se poursuit.

- AWS Organizations SCP — Le code d'application évalue les politiques de contrôle des AWS Organizations services (SCP) qui s'appliquent à la demande. Les stratégies de contrôle de service s'appliquent aux principaux du compte auquel elles sont rattachées. Si le code d'application ne trouve aucune instruction `Allow` applicable dans les SCP, le code d'application retourne la décision finale Refuser. S'il n'y a pas de SCP, ou si la SCP autorise l'action demandée, l'évaluation du code d'exécution se poursuit.
- Resource-based politiques — Au sein d'un même compte, les politiques basées sur les ressources ont un impact différent sur l'évaluation des politiques en fonction du type de principal accédant à la ressource et du principal autorisé dans la stratégie basée sur les ressources. Selon le type de principal, un `Allow` dans une politique basée sur les ressources peut aboutir à une décision finale de `Allow`, même si un rejet implicite dans une politique basée sur une identité, une limite d'autorisations ou une politique de séance est présente.

  Pour la plupart des ressources, vous n'avez besoin d'une `Allow` explicite pour le principal que dans une politique basée sur l'identité ou dans une politique basée sur les ressources pour accorder l'accès. Les politiques de confiance des rôles IAM et les politiques clés KMS constituent des exceptions à cette logique, car elles doivent autoriser explicitement l'accès pour les principaux acteurs. Resource-based politiques relatives à des services autres que l'IAM et AWS KMS peuvent également nécessiter une `Allow` déclaration explicite dans le même compte pour autoriser l'accès. Pour plus d'informations, consultez la documentation service spécifique avec lequel vous travaillez.

  Pour les demandes d'évaluation de politiques à compte unique, la logique des politiques basées sur les ressources diffère des autres types de politiques si le principal spécifié est un utilisateur IAM, un rôle IAM ou un principal de session. Les principaux de session incluent les sessions de rôle IAM ou des principaux d'utilisateur fédéréAWS STS. Si une politique basée sur les ressources accorde des autorisations directement à l'utilisateur IAM ou au principal de séance qui fait la demande, un rejet implicite dans une politique basée sur l'identité, une limite d'autorisations ou une politique de séance n'a pas d'incidence sur la décision finale.

  - Rôle IAM : Resource-based les politiques qui accordent des autorisations à un ARN de rôle IAM sont limitées par un refus implicite dans une limite d'autorisations ou une politique de session. Vous pouvez spécifier l'ARN du rôle dans l'élément Principal ou dans la clé de condition `aws:PrincipalArn`. Dans les deux cas, le principal qui fait la demande est la session de rôle IAM.

Les limites de permissions et les politiques de session ne limitent pas les permissions accordées à l'aide de la clé de condition `aws:PrincipalArn` avec un caractère générique (*) dans l'élément Principal, sauf si les politiques basées sur l'identité contiennent un refus explicite. Pour de plus amples informations, veuillez consulter Principaux de rôles IAM.

Exemple d'ARN de rôle

```
arn:aws:iam::111122223333:role/examplerole
```

- Rôle IAM – Au sein du même compte, les politiques basées sur les ressources qui accordent des autorisations à un ARN de séance de rôle IAM accordent des autorisations directement à la séance de rôle supposée. Les autorisations accordées directement à une séance ne sont pas limitées par un rejet implicite dans une politique basée sur l'identité, une limite d'autorisations ou une politique de séance. Lorsque vous endossez un rôle et faites une demande, le principal qui fait la demande est l'ARN de séance du rôle IAM et non l'ARN du rôle lui-même. Pour de plus amples informations, veuillez consulter Principaux de séance de rôle.

  Exemple d'ARN de séance de rôle

  ```
  arn:aws:sts::111122223333:assumed-role/examplerole/examplerolesessionname
  ```

- Utilisateur IAM – Au sein du même compte, les politiques basées sur les ressources qui accordent des autorisations à un ARN d'utilisateur IAM (qui n'est pas une séance d'utilisateur fédérée) ne sont pas limitées par un rejet implicite d'une politique basée sur l'identité ou d'une limite d'autorisations.

  Exemple d'ARN d'utilisateur IAM

  ```
  arn:aws:iam::111122223333:user/exampleuser
  ```

- AWS STS Federated User Principal : une session utilisateur fédérée est une session créée par un appel. `GetFederationToken` Lorsqu'un utilisateur fédéré fait une demande, le principal qui effectue la demande est l'ARN d'utilisateur fédéré et non l'ARN de l'utilisateur IAM qui a fédéré. Dans le même compte, les politiques basées sur les ressources accordant des autorisations à un ARN d'utilisateur fédéré accordent des autorisations directement à la séance. Les autorisations accordées directement à une séance ne sont pas limitées par un rejet implicite dans une politique basée sur l'identité, une limite d'autorisations ou une politique de séance.

Cependant, si une politique basée sur les ressources accorde une autorisation à l'ARN de l'utilisateur IAM qui s'est fédéré, les demandes faites par l'utilisateur fédéré pendant la séance sont limitées par un rejet implicite dans une limite d'autorisation ou une politique de séance.

Exemple d’ARN de session d’utilisateur fédérée

```
arn:aws:sts::111122223333:federated-user/exampleuser
```

- Identity-based politiques — Le code d'application vérifie les politiques basées sur l'identité pour le mandant. Pour un utilisateur IAM, cela comprend notamment les politiques d'utilisateur et les politiques de groupes auxquels l'utilisateur appartient. Si aucune politique basée sur l’identité ou aucune instruction dans les politiques basées sur l’identité n’autorise l’action demandée, alors la demande est implicitement refusée et le code d’application renvoie une décision finale Refuser. Si une instruction dans n’importe quelle politique basée sur l’identité applicable autorise l’action demandée, le code d’évaluation continue.
- Limite des autorisations IAM : le code d’application vérifie si l’entité IAM utilisée par le principal possède une limite d’autorisations. Si la politique qui est utilisée pour définir la limite d'autorisations n'autorise pas l'action demandée, la demande est implicitement refusée. Le code d'application retourne la décision finale Deny (Refuser). S’il n’y a pas de limite d’autorisations, ou si la limite d’autorisations autorise l’action demandée, le code d’évaluation continue.
- Politiques de session : le code d’application vérifie si le principal est un principal de session. Les principaux de session incluent une session de rôle IAM ou une session utilisateur AWS STS fédérée. Si le principal n'est pas un principal de séance, le code d'application renvoie une décision finale d'autorisation.

  Pour les principaux de séance, le code d’application vérifie si une politique de séance a été transmise dans la demande. Vous pouvez transmettre une politique de session tout en utilisant l' AWS API AWS CLI or pour obtenir des informations d'identification temporaires pour un rôle ou un utilisateur principal AWS STS fédéré. Si vous n’avez pas adopté de politique de session, une politique de session par défaut est créée et le code d’application renvoie la décision finale Autoriser.

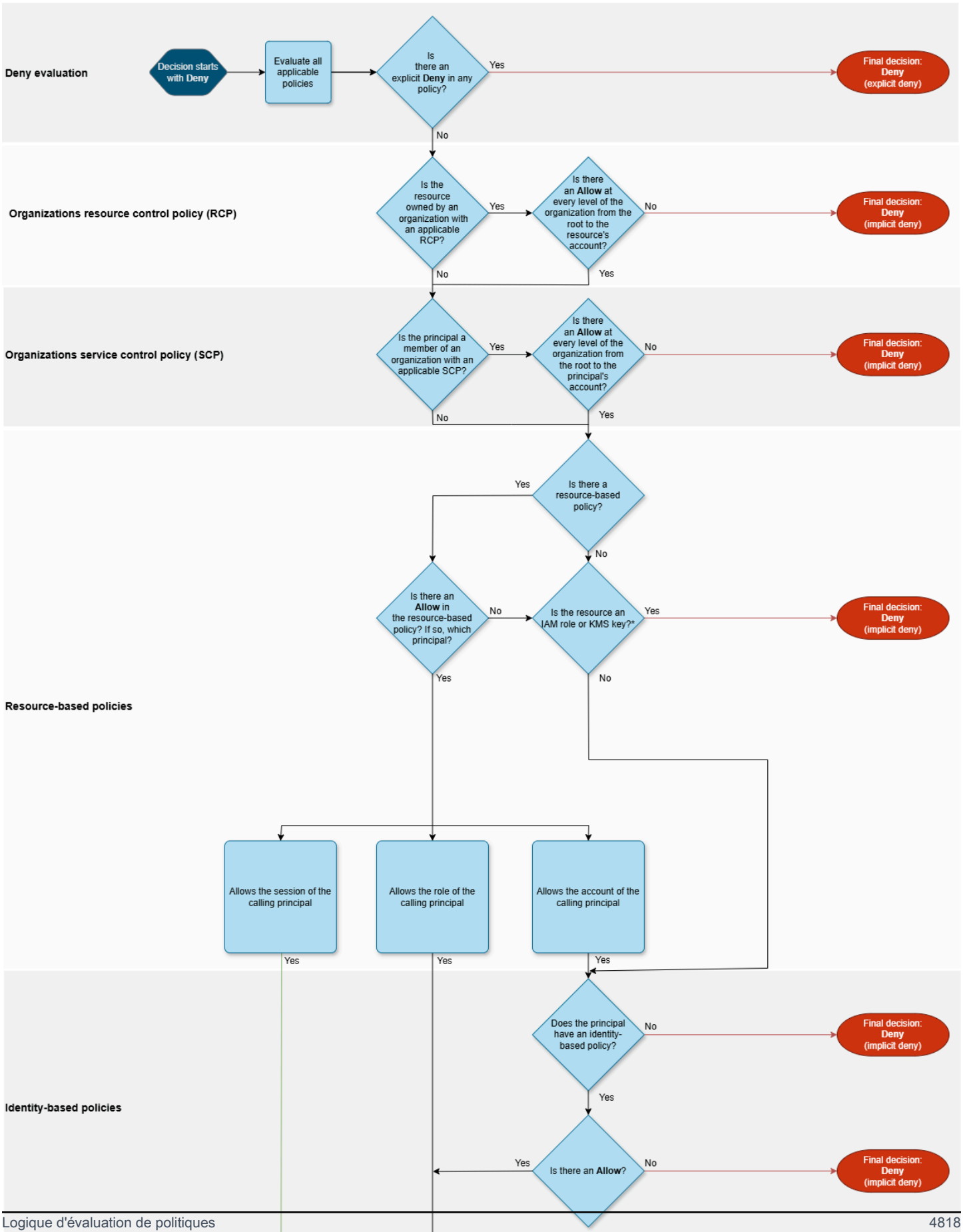
  - Si une politique de session est présente et n'autorise pas l'action demandée, la demande est implicitement refusée. Le code d'application retourne la décision finale Deny (Refuser).
  - Le code d’application vérifie si le principal est une session de rôle. Si le principal est une session de rôle, la demande est autorisée. Sinon, la demande est implicitement rejetée et le code d’application renvoie une décision finale Refuser.

- Si une politique de séance est présente et autorise l'action demandée, le code d'exécution renvoie une décision finale d'autorisation.

Évaluation des politiques pour les demandes au sein d’un même compte

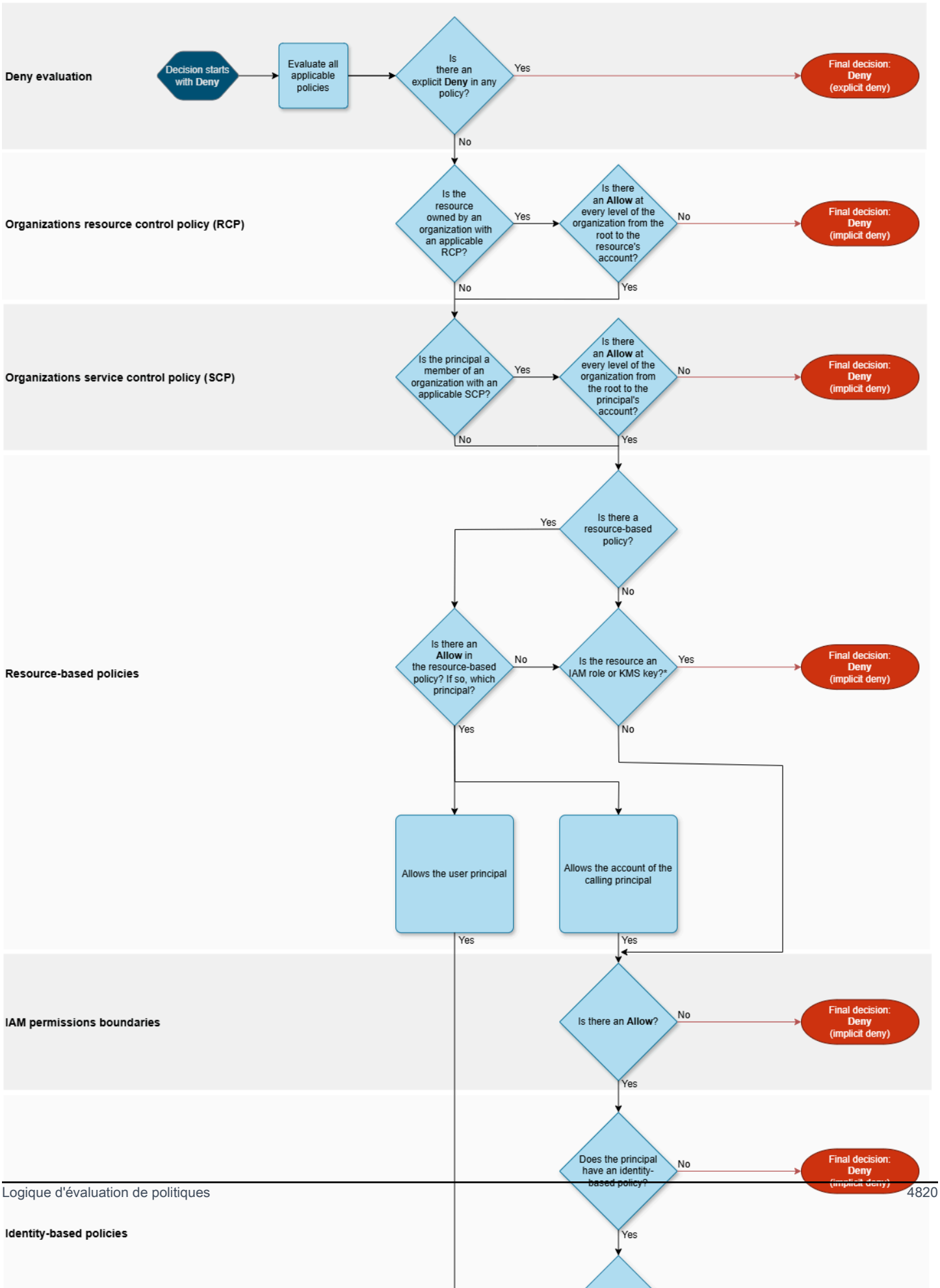
Évaluation des politiques pour un rôle IAM

Le diagramme suivant fournit des détails sur la manière dont une décision d’évaluation de politique est prise pour un rôle IAM au sein d’un compte unique.


Deny evaluation
Decision starts with Deny
Evaluate all applicable policies
Is there an explicit Deny in any policy?
Yes
Final decision: Deny (explicit deny)
No
Organizations resource control policy (RCP)
Is the resource owned by an organization with an applicable RCP?
Yes
Is there an Allow at every level of the organization from the root to the resource's account?
No
Final decision: Deny (implicit deny)
No
Yes
Organizations service control policy (SCP)
Is the principal a member of an organization with an applicable SCP?
Yes
Is there an Allow at every level of the organization from the root to the principal's account?
No
Final decision: Deny (implicit deny)
No
Yes
Resource-based policies
Yes
Is there a resource-based policy?
No
Is there an Allow in the resource-based policy? If so, which principal?
No
Is the resource an IAM role or KMS key?*
Yes
Final decision: Deny (implicit deny)
Yes
No
Allows the session of the calling principal
Allows the role of the calling principal
Allows the account of the calling principal
Yes
Yes
Yes
Identity-based policies
Does the principal have an identity-based policy?
No
Final decision: Deny (implicit deny)
Yes
Yes
Is there an Allow?
No
Final decision: Deny (implicit deny)
IAM permissions boundaries
Does the principal have a permissions
Yes
Is there an Allow?
No
Final decision: Deny

Évaluation des politiques pour un utilisateur IAM

Le diagramme suivant fournit des détails sur la manière dont une décision d’évaluation de politique est prise pour un utilisateur IAM au sein d’un compte unique.


Deny evaluation
Decision starts with Deny
Evaluate all applicable policies
Is there an explicit Deny in any policy?
Yes
Final decision: Deny (explicit deny)
No
Organizations resource control policy (RCP)
Is the resource owned by an organization with an applicable RCP?
Yes
Is there an Allow at every level of the organization from the root to the resource's account?
No
Final decision: Deny (implicit deny)
No
Yes
Organizations service control policy (SCP)
Is the principal a member of an organization with an applicable SCP?
Yes
Is there an Allow at every level of the organization from the root to the principal's account?
No
Final decision: Deny (implicit deny)
No
Yes
Resource-based policies
Is there a resource-based policy?
Yes
No
Is there an Allow in the resource-based policy? If so, which principal?
No
Is the resource an IAM role or KMS key?*
Yes
Final decision: Deny (implicit deny)
Yes
No
Allows the user principal
Allows the account of the calling principal
Yes
Yes
IAM permissions boundaries
Is there an Allow?
No
Final decision: Deny (implicit deny)
Yes
Does the principal have an identity-based policy?
No
Final decision: Deny (implicit deny)
Yes
Identity-based policies

Exemple d'évaluation de politique basée sur l'identité et sur les ressources

Les types de politiques les plus courants sont celles basées sur l'identité et les ressources. Lorsque l'accès à une ressource est demandé, AWS évalue toutes les autorisations accordées par les politiques pour au moins une autorisation au sein du même compte. Un rejet explicite dans l'une de ces politiques remplace l'autorisation.

> **Important**
>
> Si la politique basée sur les identités ou celle basée sur les ressources dans le même compte autorise la demande et que l'autre ne l'autorise pas, la demande reste autorisée.

Supposons que le nom d'utilisateur de Carlos soit `carlossalazar` et que ce dernier essaie d'enregistrer un fichier dans le compartiment Amazon S3 `amzn-s3-demo-bucket-carlossalazar-logs`.

Supposons également que la politique suivante soit attachée à l'utilisateur IAM `carlossalazar`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowS3Self",
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar/*",
```

```
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar"
            ]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::*log*"
        }
    ]
}
```

L'instruction `AllowS3ListRead` de cette politique autorise Carlos à afficher une liste de tous les compartiments du compte. L'instruction `AllowS3Self` accorde à Carlos un accès total au compartiment du même nom que son nom d'utilisateur. L'instruction `DenyS3Logs` refuse à Carlos l'accès à n'importe quel compartiment S3 comprenant `log` dans son nom.

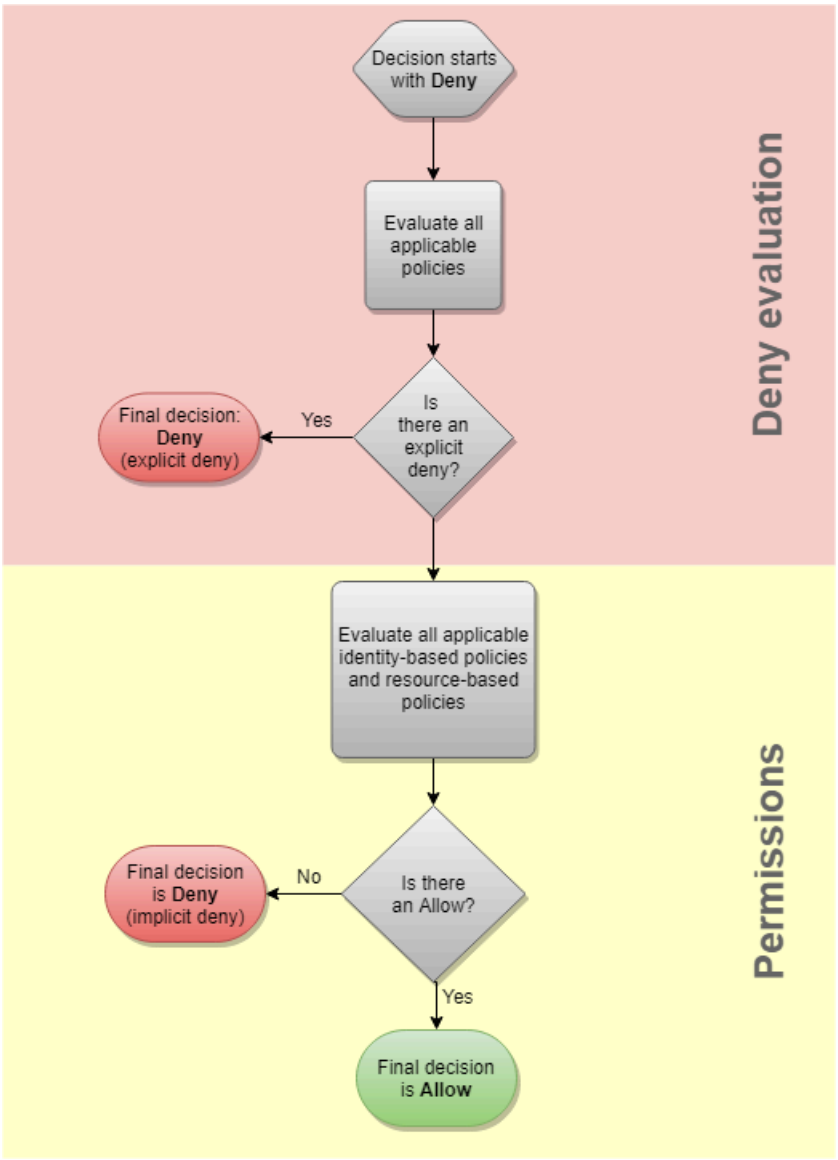
De plus, la politique basée sur les ressources suivante (appelée politique de compartiment) est attachée au compartiment `amzn-s3-demo-bucket-carlossalazar`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/carlossalazar"
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar/*",
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar"
            ]
        }
    ]
}
```

Cette politique spécifie que seul l'utilisateur `carlossalazar` peut accéder au compartiment `amzn-s3-demo-bucket-carlossalazar`.

Lorsque Carlos demande l'enregistrement d'un fichier dans le `amzn-s3-demo-bucket-carlossalazar-logs` bucket, il AWS détermine les politiques qui s'appliquent à la demande. Dans ce cas, seule les politiques basées sur l'identité et les ressources s'appliquent. Il s'agit de deux politiques d'autorisations. La logique d'évaluation est réduite à la logique suivante, car aucune limite d'autorisations ne s'applique.



AWS vérifie d'abord si une `Deny` déclaration s'applique au contexte de la demande. Il en trouve une, car la politique basée sur l'identité refuse explicitement à Carlos l'accès à n'importe quel compartiment S3 utilisé pour la journalisation. Carlos se voit refuser l'accès.

Supposons qu'il se rende compte de son erreur et essaie d'enregistrer le fichier dans le `amzn-s3-demo-bucket-carlossalazar` bucket. AWS recherche une `Deny` déclaration et n'en trouve aucune. Ensuite, il vérifie les politiques d'autorisations. La politique basée sur l'identité et celle basée sur les ressources autorisent la demande. AWS Autorise donc la demande. Si l'un d'entre eux refuse explicitement l'instruction, alors la demande est refusée. Si l'un des types de politique autorise la demande et que l'autre ne l'autorise pas, la demande reste autorisée.

Cross-account logique d'évaluation des politiques

Vous pouvez autoriser un principal d'un compte à accéder aux ressources d'un second compte. C'est ce que l'on appelle l'accès entre comptes. Lorsque vous autorisez un accès entre comptes, le compte dans lequel se trouve le principal est appelé compte approuvé. Le compte dans lequel se trouve la ressource est le compte d'approbation.

Pour autoriser l'accès entre comptes, vous associez une politique basée sur les ressources à la ressource que vous souhaitez partager. Vous devez également attacher une politique basée sur l'identité à l'identité qui fait office de principal dans la demande. La politique basée sur les ressources dans le compte d'approbation doit spécifier le principal du compte approuvé qui aura accès à la ressource. Vous pouvez spécifier l'intégralité du compte ou ses utilisateurs IAM, ses utilisateurs principaux AWS STS fédérés, ses rôles IAM ou ses sessions à rôles assumés. Vous pouvez également spécifier un AWS service en tant que principal. Pour de plus amples informations, veuillez consulter Comment spécifier un principal.

La politique basée sur l'identité du principal doit permettre l'accès demandé à la ressource dans le service d'approbation. Vous pouvez le faire en spécifiant l'ARN de la ressource.

Dans IAM, vous pouvez associer une politique basée sur les ressources à un rôle IAM pour permettre aux principaux d'autres comptes d'endosser ce rôle. La politique basée sur les ressources du rôle est appelée politique d'approbation de rôle. Une fois qu'ils endossent ce rôle, les principaux autorisés peuvent utiliser les informations d'identification temporaires résultantes pour accéder à plusieurs ressources de votre compte. Cet accès est défini dans la politique d'autorisations basée sur l'identité du rôle. Pour connaître les différences entre l'autorisation d'accès entre comptes à l'aide de rôles et l'autorisation d'accès entre comptes à l'aide d'autres politiques basées sur les ressources, veuillez consulter Accès intercompte aux ressources dans IAM.

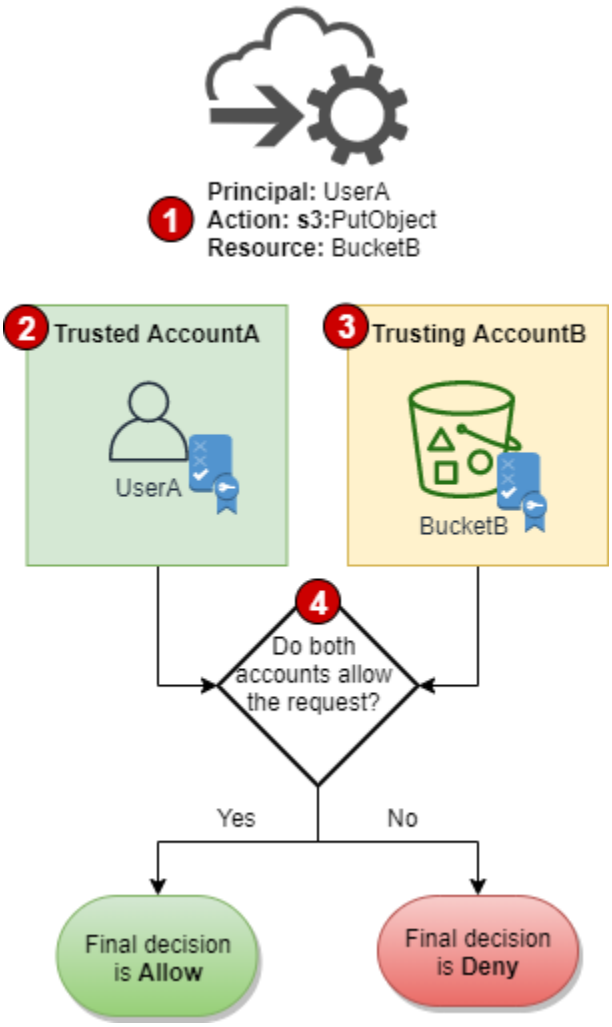
> **Important**
>
> D'autres services peuvent influer sur la logique d'évaluation des politiques. Par exemple, AWS Organizations prend en charge les politiques de contrôle des services et les politiques

de contrôle des ressources qui peuvent être appliquées aux principaux et aux ressources d'un ou de plusieurs comptes. AWS Resource Access Manager prend en charge les fragments de politique qui contrôlent les actions que les principaux sont autorisés à effectuer sur les ressources partagées avec eux.

Comment déterminer si une demande d'accès entre comptes est autorisée

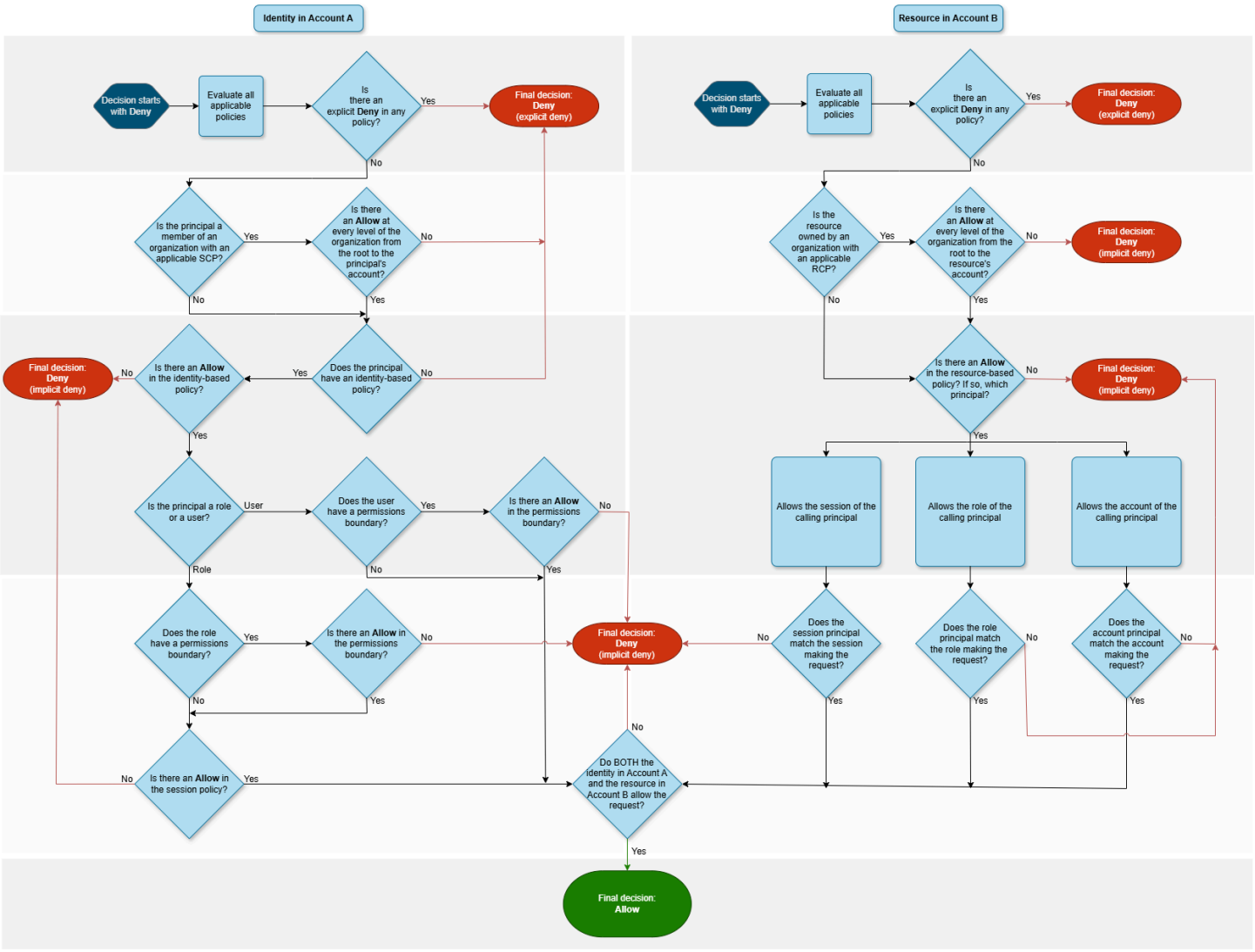
Pour les demandes entre comptes, le demandeur du compte approuvé `AccountA` doit avoir une politique basée sur l'identité. Cette politique doit lui permettre de faire une demande à la ressource du compte d'approbation `AccountB`. En outre, la politique basée sur les ressources du compte `AccountB` doit autoriser le demandeur du compte `AccountA` à accéder à la ressource.

Lorsque vous faites une demande entre comptes, AWS effectue deux évaluations. AWS évalue la demande dans le compte de confiance et le compte de confiance. Pour de plus amples informations sur la façon dont une demande est évaluée au sein d'un seul compte, veuillez consulter Comment ? AWS la logique du code d'application évalue les demandes pour autoriser ou refuser l'accès. La demande n'est autorisée que si les deux évaluations renvoient une décision `Allow`.



1. Lorsqu'un principal d'un compte fait une demande d'accès à une ressource d'un autre compte, il s'agit d'une demande entre comptes.
2. Le principal demandeur se trouve dans le compte approuvé (`AccountA`). Lorsque AWS évalue ce compte, il vérifie la politique basée sur l'identité et toutes les politiques pouvant limiter une politique basée sur l'identité. Pour de plus amples informations, veuillez consulter Évaluation des politiques basées sur l'identité avec des limites d'autorisations.
3. La ressource demandée se trouve dans le compte d'approbation (`AccountB`). Lorsque AWS évalue ce compte, il vérifie la politique basée sur les ressources qui est associée à la ressource demandée et toutes les politiques pouvant limiter une politique basée sur les ressources. Pour de plus amples informations, veuillez consulter Évaluation des politiques basées sur l'identité avec des politiques basées sur des ressources.
4. AWS autorise la demande uniquement si les deux évaluations de la politique du compte l'autorisent.

Le diagramme suivant illustre plus en détail comment une décision d'évaluation de politique est prise pour une demande entre comptes. Encore une fois, AWS autorise la demande uniquement si les deux évaluations de la politique du compte l'autorisent.



Exemple d'évaluation de politique entre comptes

L'exemple suivant illustre un scénario dans lequel un rôle d'un compte se voit accorder des autorisations par une politique basée sur les ressources dans un second compte.

Supposons que Carlos est un développeur dont le nom du rôle IAM est `Demo` dans le compte 111111111111. Il veut enregistrer un fichier dans le compartiment Amazon S3 `amzn-s3-demo-bucket-production-logs` du compte 222222222222.

Supposons également que la politique suivante soit attachée au rôle IAM `Demo`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "*"
        },
        {
            "Sid": "AllowS3ProductionObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object*",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-production/*"
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::*log*",
                "arn:aws:s3:::*log*/*"
            ]
        }
    ]
}
```

L'instruction `AllowS3ListRead` de cette politique autorise Carlos à afficher une liste de tous les compartiments dans Amazon S3. L'instruction `AllowS3ProductionObjectActions` offre à Carlos un accès complet aux objets dans le compartiment `amzn-s3-demo-bucket-production`.

De plus, la politique basée sur les ressources suivante (appelée politique de compartiment) est attachée au compartiment `amzn-s3-demo-bucket-production` dans le compte 222222222222.

JSON

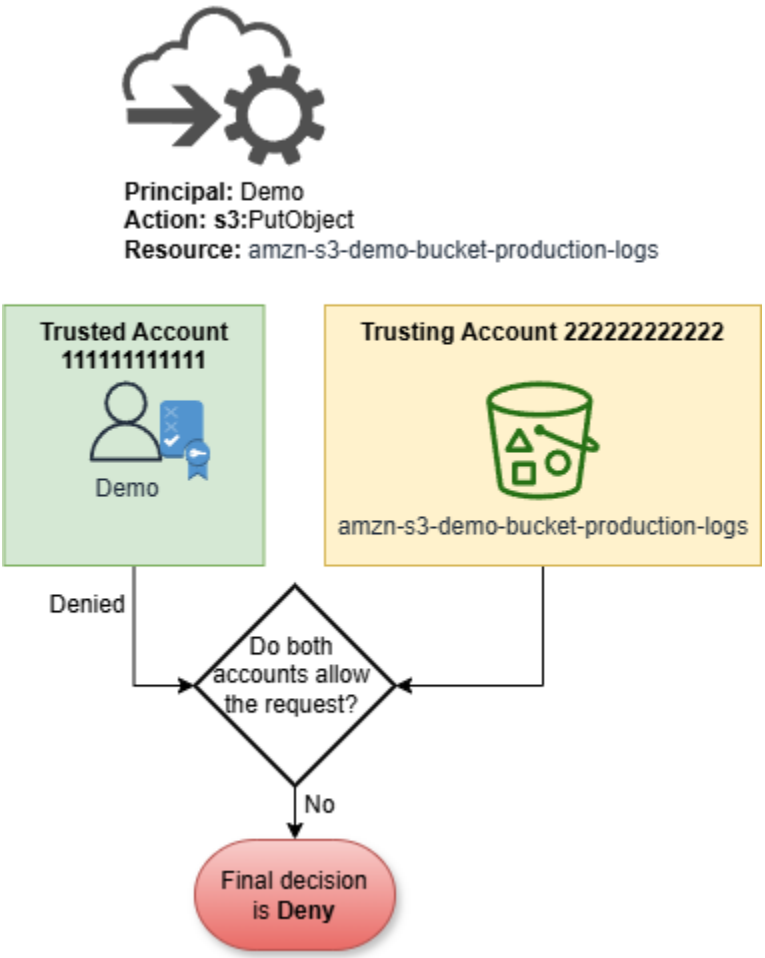
```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject*",
                "s3:PutObject*",
                "s3:ReplicateObject",
                "s3:RestoreObject"
            ],
            "Principal": { "AWS": "arn:aws:iam::111111111111:role/Demo" },
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-production/*"
        }
    ]
}
```

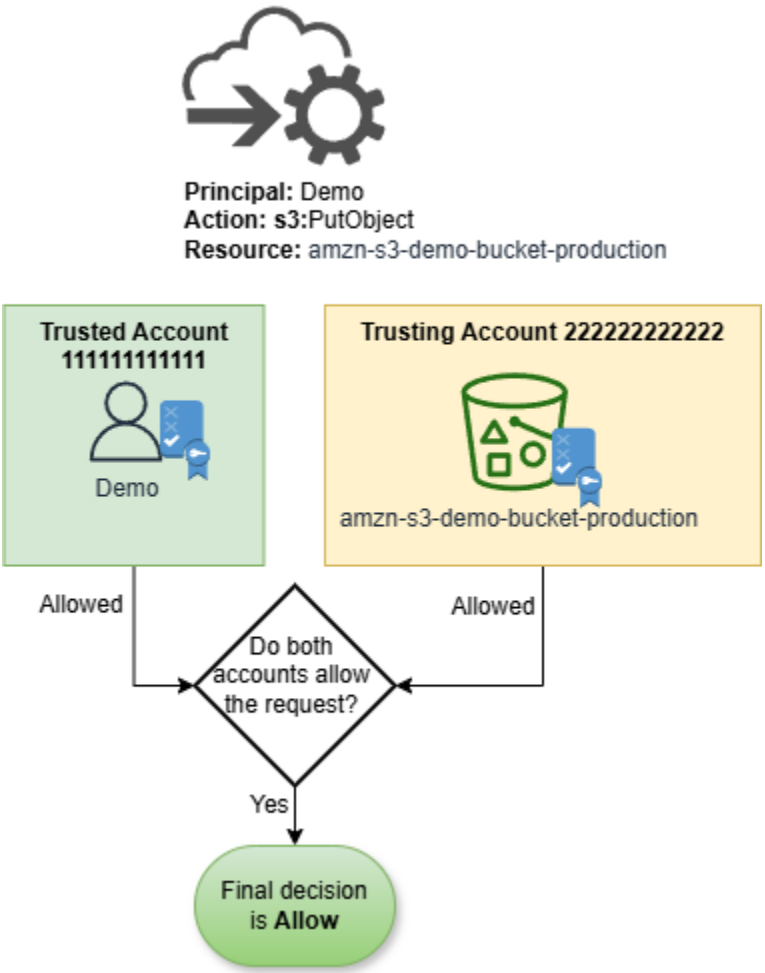
Cette politique permet au rôle `Demo` d'accéder aux objets du compartiment `amzn-s3-demo-bucket-production`. Il peut créer et éditer, mais pas supprimer les objets dans le compartiment. Il ne peut pas gérer le compartiment lui-même.

Lorsque Carlos demande l'enregistrement d'un fichier dans le `amzn-s3-demo-bucket-production-logs` bucket, il AWS détermine les politiques qui s'appliquent à la demande. Dans ce cas, la politique basée sur l'identité associée au rôle `Demo` est la seule politique qui s'applique dans le compte `111111111111`. Dans le compte `222222222222`, il n'y a pas de politique basée sur les ressources associée au compartiment `amzn-s3-demo-bucket-production-logs`. Lors de AWS l'évaluation du compte`111111111111`, il renvoie une décision de`Deny`. En effet, l'instruction `DenyS3Logs` de la politique basée sur l'identité refuse explicitement l'accès à tous les compartiments de journaux. Pour de plus amples informations sur la façon dont une demande est évaluée au sein d'un seul compte, veuillez consulter Comment ? AWS la logique du code d'application évalue les demandes pour autoriser ou refuser l'accès.

Étant donné que la demande est explicitement refusée dans l'un des comptes, la décision finale est de refuser la demande.



Supposons que Carlos réalise alors son erreur et essaie d'enregistrer le fichier dans le `Production` bucket. AWS vérifie d'abord `111111111111` le compte pour déterminer si la demande est autorisée. Seule la politique basée sur l'identité s'applique et autorise la demande. AWS puis vérifie le compte`222222222222`. Seule la politique basée sur les ressources associée au compartiment `Production` s'applique et elle autorise la demande. Étant donné que les deux comptes autorisent la demande, la décision finale est d'autoriser la demande.



Différence entre les refus explicites et implicites

Une demande se traduit par un refus explicite si une politique applicable inclut une instruction `Deny` (Refuser). Si les politiques qui s'appliquent à une demande incluent une instruction `Allow` (Autoriser) et une instruction `Deny` (Refuser), alors l'instruction `Deny` a priorité sur l'instruction `Allow` (Autoriser). La demande est refusée explicitement.

Un refus implicite se produit en cas d'absence d'instructions `Deny` (Refuser) et `Allow` (Autoriser) applicables. Étant donné qu'un principal IAM se voit refuser l'accès par défaut, il doit être explicitement autorisé à effectuer une action. Sinon, il se voit refuser l'accès implicitement.

Lorsque vous concevez votre stratégie d'autorisations, vous devez créer des politiques avec des instructions `Allow` (Autoriser) pour permettre à vos principaux d'effectuer des demandes avec succès. Toutefois, vous pouvez choisir n'importe quelle combinaison de refus explicites et implicites.

Par exemple, vous pouvez créer la politique suivante qui inclut des actions autorisées, des actions implicitement rejetées et des actions explicitement rejetées. L'instruction `AllowGetList`

permet un accès en lecture seule aux actions IAM qui commencent par les préfixes `Get` et `List`. Toutes les autres actions dans IAM, comme `iam:CreatePolicy`, sont implicitement rejetées. L'instruction `DenyReports` rejette explicitement l'accès aux rapports IAM en rejetant l'accès aux actions qui incluent le suffixe `Report`, comme `iam:GetOrganizationsAccessReport`. Si quelqu'un ajoute une autre politique à ce principal pour lui donner accès aux rapports IAM, comme `iam:GenerateCredentialReport`, les demandes liées aux rapports sont toujours rejetées en raison de ce rejet explicite.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowGetList",
            "Effect": "Allow",
            "Action": [
                "iam:Get*",
                "iam:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "DenyReports",
            "Effect": "Deny",
            "Action": "iam:*Report",
            "Resource": "*"
        }
    ]
}
```

Syntaxe du langage de politique JSON IAM

Cette page présente la syntaxe formelle du langage de stratégie utilisé pour créer des politiques JSON dans IAM. Nous vous la présentons pour vous permettre de mieux comprendre comment créer et valider des politiques.

Pour consulter des exemples de politiques, reportez-vous aux rubriques suivantes :

- Politiques et autorisations dans Gestion des identités et des accès AWS

- Exemples de politiques basées sur l'identité IAM
- Exemples de politiques pour travailler dans la console Amazon EC2 et exemples de politiques pour travailler avec la AWS CLI, la CLI Amazon EC2 ou un AWS SDK dans le guide de l'utilisateur Amazon EC2.
- Exemples de politique de compartiment et Exemples de politiques utilisateur dans le guide de l'utilisateur service de stockage simple Amazon.

Pour obtenir des exemples de politiques utilisées dans d'autres AWS services, consultez la documentation de ces services.

Rubriques

- Langage de politique et JSON
- Conventions utilisées dans cette syntaxe
- Syntaxe
- Remarques sur la syntaxe de politique

Langage de politique et JSON

Les politiques sont créées à l'aide du format JSON. Lorsque vous créez ou modifiez une politique JSON, IAM peut effectuer une validation de politique pour vous aider à créer une politique efficace. IAM identifie les erreurs de syntaxe JSON, tandis que IAM Access Analyzer fournit des vérifications de politique supplémentaires avec des recommandations pour vous aider à affiner vos politiques. Pour en savoir plus sur la validation de politiques, veuillez consulter Validation de politique IAM. Pour en savoir plus sur les vérifications des politiques IAM Access Analyzer et les recommandations exploitables, veuillez consulter Validation de politique IAM Access Analyzer.

Ce document n'entend pas fournir une description complète de ce qui constitue un code JSON valide. Toutefois, voici quelques règles de base applicables à JSON :

- Les espaces entre des entités individuelles sont autorisés.
- Les valeurs sont entourées de guillemets. Les guillemets sont facultatifs pour les valeurs numériques et booléennes.
- De nombreux éléments (par exemple, `action_string_list` et `resource_string_list`) acceptent une valeur composée d'un tableau JSON. Les tableaux peuvent inclure une ou plusieurs valeurs. Lorsqu'un tableau comporte plusieurs valeurs, il est entouré de crochets (`[` et `]`) et les valeurs sont séparées par une virgule, comme dans l'exemple suivant :

```
"Action" : ["ec2:Describe*","ec2:List*"]
```

- Les types de données JSON de base (booléen, nombre et chaîne) sont définis dans RFC 7159.

Conventions utilisées dans cette syntaxe

Les conventions suivantes sont utilisées dans cette syntaxe :

- Les caractères suivants sont des jetons JSON et sont inclus dans les politiques :

  ```
  { } [ ] " , :
  ```

- Les caractères suivants sont des caractères spéciaux dans la syntaxe et ne sont pas inclus dans les politiques :

  ```
  = < > ( ) |
  ```

- Si un élément autorise plusieurs valeurs, ceci est indiqué à l'aide de valeurs répétées, d'une virgule de délimitation et de points de suspension (`...`). Exemples :

  ```
  [<action_string>, <action_string>, ...]
  ```

  ```
  <principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }
  ```

  Si plusieurs valeurs sont autorisées, il est également possible d'en inclure une seule. Lors de l'utilisation d'une seule valeur, la virgule de fin doit être omise. Si l'élément accepte un tableau (entouré de [et]) mais que vous n'utilisez qu'une valeur, les crochets sont facultatifs. Exemples :

  ```
  "Action": [<action_string>]
  ```

  ```
  "Action": <action_string>
  ```

- Un point d'interrogation (?) placé après un élément indique que cet élément est facultatif. Exemple :

  ```
  <version_block?>
  ```

  Toutefois, reportez-vous aux notes relatives à la syntaxe ci-après pour plus de détails sur les éléments facultatifs.

- Une ligne verticale (|) entre des éléments indique plusieurs choix. Dans la syntaxe, les parenthèses définissent la portée de ces choix. Exemple :

```
("Principal" | "NotPrincipal")
```

- Les éléments qui doivent être des chaînes littérales sont entourés de guillemets doubles ("). Exemple :

```
<version_block> = "Version" : ("2008-10-17" | "2012-10-17" )
```

Pour des informations complémentaires, reportez-vous à la section Remarques sur la syntaxe de politique après la description de la syntaxe.

Syntaxe

La liste suivante décrit la syntaxe du langage de politique. Pour plus d'informations sur les conventions utilisées dans la liste, reportez-vous à la section précédente. Pour des informations complémentaires, reportez-vous aux notes ci-après.

> **Note**
>
> Cette syntaxe décrit des politiques marquées avec une version `2008-10-17` et `2012-10-17`. Un élément de politique `Version` varie d'une version de politique. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. En revanche, une version de politique est créée lorsque vous apportez des modifications à une politique gérée par le client dans IAM. La politique modifiée ne remplace pas la politique existante. À la place, IAM crée une nouvelle version de la politique gérée. Pour en savoir plus sur l'élément de politique `Version`, consultez Éléments de politique JSON IAM : Version. Pour en savoir plus sur les versions de politiques, consultez the section called "Gestion des versions des politiques IAM".

```
policy  = {
     <version_block?>,
     <id_block?>,
     <statement_block>
}

<version_block> = "Version" : ("2008-10-17"        | "2012-10-17"       )

<id_block> = "Id" : <policy_id_string>
```

```
<statement_block> = "Statement" : [ <statement>, <statement>, ... ]

<statement> = {
    <sid_block?>,
    <principal_block?>,
    <effect_block>,
    <action_block>,
    <resource_block>,
    <condition_block?>
}

<sid_block> = "Sid" : <sid_string>

<effect_block> = "Effect" : ("Allow" | "Deny")

<principal_block> = ("Principal" | "NotPrincipal") : ("*" | <principal_map>)

<principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }

<principal_map_entry> = ("AWS" | "Federated" | "Service" | "CanonicalUser") :
    [<principal_id_string>, <principal_id_string>, ...]

<action_block> = ("Action" | "NotAction") :
    ("*" | <action_string> | [<action_string>, <action_string>, ...])

<resource_block> = ("Resource" | "NotResource") :
    : ("*" | <resource_string> | [<resource_string>, <resource_string>, ...])

<condition_block> = "Condition" : { <condition_map> }
<condition_map> = {
  <condition_type_string> : { <condition_key_string> : <condition_value_list> },
  <condition_type_string> : { <condition_key_string> : <condition_value_list> }, ...
}
<condition_value_list> = [<condition_value>, <condition_value>, ...]
<condition_value> = (<condition_value_string> | <condition_value_string> |
 <condition_value_string>)
```

Remarques sur la syntaxe de politique

- Une politique peut contenir un tableau de instructions.

- La taille maximale des politiques est comprise entre 2 048 et 10 240 caractères, selon l'entité à laquelle la politique est attachée. Pour plus d'informations, veuillez consulter IAM et AWS STS quotas. Lors du calcul de la taille de la politique, les espaces ne sont pas inclus.
- Des éléments individuels ne doivent pas contenir plusieurs instances de la même clé. Par exemple, vous ne pouvez pas inclure le bloc `Effect` à deux reprises dans la même instruction.
- Les blocs peuvent être insérés dans n'importe quel ordre. Par exemple, `version_block` peut suivre `id_block` dans une politique. De la même façon `effect_block`, `principal_block` et `action_block` peuvent figurer dans n'importe quel ordre dans une instruction.
- L'élément `id_block` est facultatif dans les politiques basées sur les ressources. Il ne doit pas être inclus dans les stratégies basées sur une identité.
- L'élément `principal_block` est obligatoire dans les politiques basées sur les ressources (par exemple, dans les politiques de compartiment Amazon S3) et dans les politiques d'approbation de rôles IAM. Il ne doit pas être inclus dans les stratégies basées sur une identité.
- L'élément `principal_map` des politiques de compartiment Amazon S3 peut inclure l'ID `CanonicalUser`. La plupart des politiques basées sur les ressources ne prennent pas en charge ce mappage. Pour en savoir plus sur l'utilisation de l'ID d'utilisateur canonique dans une politique de compartiment, veuillez consulter spécification d'un principal dans une politique dans le guide de l'utilisateur service de stockage simple Amazon.
- Chaque valeur de chaîne (`policy_id_string`, `sid_string`, `principal_id_string`, `action_string`, `resource_string`, `condition_type_string`, `condition_key_string` et la version de chaîne de `condition_value`) peut avoir ses propres limites de longueur minimale et maximale, des valeurs autorisées spécifiques ou un format interne obligatoire.

Remarques à propos de valeurs de chaîne

Cette section contient des informations complémentaires sur les valeurs de chaîne utilisées dans les différents éléments d'une politique.

`action_string`

Comporte un espace de noms de service, deux points et le nom d'une action. Un nom d'action peut inclure des caractères génériques. Exemples :

```
"Action":"ec2:StartInstances"

"Action":[
  "ec2:StartInstances",
```

```
  "ec2:StopInstances"
]

"Action":"cloudformation:*"

"Action":"*"

"Action":[
  "s3:Get*",
  "s3:List*"
]
```

policy_id_string

Permet d'inclure des informations globales sur la politique. Certains services tels qu'Amazon SQS et Amazon SNS utilisent l'élément `Id` en tant qu'élément réservé. Si aucune restriction n'est imposée par un service individuel, policy_id_string peut inclure des espaces. Certains services exigent que cette valeur soit unique dans un compte AWS .

> **Note**
>
> L'élément `id_block` est autorisé dans les politiques basées sur les ressources, mais pas dans celles basées sur une identité.

La longueur n'est pas limitée, mais cette chaîne contribue à la longueur totale de la politique, qui elle est limitée.

```
"Id":"Admin_Policy"

"Id":"cd3ad3d9-2776-4ef1-a904-4c229d1642ee"
```

sid_string

Permet d'inclure des informations relatives à une instruction individuelle. Pour les politiques IAM, les caractères alphanumériques de base (A-Z, a-z,0-9) sont les seuls caractères autorisés dans la valeur. `Sid` Les autres services AWS prenant en charge les politiques de ressources peuvent imposer d'autres exigences pour la valeur `Sid`. Par exemple, certains services exigent que cette valeur soit unique au sein d'un Compte AWS, et certains services autorisent des caractères supplémentaires tels que des espaces dans la `Sid` valeur.

```
"Sid":"1"

"Sid": "ThisStatementProvidesPermissionsForConsoleAccess"
```

`principal_id_string`

Permet de spécifier un principal à l'aide du nom de ressource Amazon (ARN) de l'utilisateur IAM Compte AWS, du rôle IAM, de l'utilisateur fédéré ou de l'utilisateur ayant un rôle assumé. Pour un Compte AWS, vous pouvez également utiliser la forme abrégée `AWS:accountnumber` au lieu de l'ARN complet. Pour toutes les options, notamment les services AWS ;, les rôles endossés et ainsi de suite, consultez Comment spécifier un principal.

Notez que vous ne pouvez utiliser * que pour spécifier «everyone/anonymous. » Vous ne pouvez pas l'utiliser pour spécifier une partie d'un nom ou d'un ARN.

`resource_string`

Dans la plupart des cas, la valeur est composée d'un Amazon Resource Name (ARN). Vous pouvez utiliser des caractères génériques (* et ?) dans la partie ressource de l'ARN. Pour plus d'informations sur l'utilisation des caractères génériques dans les ARN, consultez Utilisation de caractères génériques dans les chemins.

> **Note**
>
> Lorsque vous spécifiez un ARN incomplet (un contenant moins de six champs standard) dans une politique basée sur l'identité, complète AWS automatiquement l'ARN en ajoutant des caractères génériques (*) à tous les champs manquants. Par exemple, spécifier `arn:aws:sqs` est équivalent à`arn:aws:sqs:*:*:*`, ce qui donne accès à toutes les ressources Amazon Amazon SQS dans toutes les régions et tous les comptes. Cependant, les politiques de session transmises et AWS STS AssumeRole AssumeRoleWithWebIdentity les AssumeRoleWithSAML demandes ne prennent pas en charge les ARN incomplets. L'utilisation d'un ARN incomplet dans une politique de session entraînera une `MalformedPolicyDocumentException` erreur.

```
"Resource":"arn:aws:iam::123456789012:user/Bob"

"Resource":"arn:aws:s3:::amzn-s3-demo-bucket/*"
```

`condition_type_string`

Identifie le type de condition testé, par exemple `StringEquals`, `StringLike`, `NumericLessThan`, `DateGreaterThanEquals`, `Bool`, `BinaryEquals`, `IpAddress`, `ArnEquals`, etc. Pour obtenir la liste complète des types de conditions, consultez Éléments de politique JSON IAM : Opérateurs de condition.

```
"Condition": {
  "NumericLessThanEquals": {
    "s3:max-keys": "10"
  }
}

"Condition": {
  "Bool": {
    "aws:SecureTransport": "true"
  }
}

"Condition": {
  "StringEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
}
```

`condition_key_string`

Identifie la clé de condition dont la valeur sera testée pour déterminer si la condition est remplie. AWS définit un ensemble de clés de condition disponibles dans tous les AWS services, y compris `aws:PrincipalTypeaws:SecureTransport`, et`aws:userid`.

Pour obtenir la liste des clés de AWS condition, consultezAWS clés de contexte de condition globale. Pour connaître les clés de condition spécifiques à un service, reportez-vous à la documentation relative au service, notamment :

- Spécification de conditions dans une politique dans le guide de l'utilisateur service de stockage simple Amazon
- Politiques IAM pour Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.

```
"Condition":{
  "Bool": {
```

```
      "aws:SecureTransport": "true"
   }
}

"Condition": {
  "StringNotEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
}

"Condition": {
  "StringEquals": {
    "aws:ResourceTag/purpose": "test"
  }
}
```

`condition_value_string`

Identifie la valeur de la chaîne condition_key_string qui détermine si la condition est remplie. Pour obtenir la liste complète des valeurs valides pour un type de condition, consultez Éléments de politique JSON IAM : Opérateurs de condition.

```
"Condition":{
  "ForAnyValue:StringEquals": {
  "dynamodb:Attributes": [
   "ID",
   "PostDateTime"
         ]
  }
}
```

AWS politiques gérées pour les fonctions professionnelles

Nous vous recommandons d'utiliser des politiques qui accordent le moins de privilèges ou d'accorder uniquement les autorisations requises pour effectuer une tâche. Le moyen le plus sûr d'octroyer le moindre privilège consiste à écrire une politique personnalisée contenant uniquement les autorisations requises par votre équipe. Vous devez créer un processus pour autoriser votre équipe à demander plus d'autorisations si nécessaire. Il faut du temps et de l'expertise pour créer des politiques gérées par le client IAM qui ne fournissent à votre équipe que les autorisations dont elle a besoin.

Pour commencer à ajouter des autorisations à vos identités IAM (utilisateurs, groupes d'utilisateurs et rôles), vous pouvez utiliserAWS stratégies gérées. AWS les politiques gérées couvrent les cas d'utilisation courants et sont disponibles dans votre Compte AWS. AWS les politiques gérées n'accordent pas les autorisations de moindre privilège. Vous devez prendre en compte le risque de sécurité constitué par l'octroi, à vos principaux, de davantage d'autorisations que nécessaire pour accomplir leur tâche.

Vous pouvez associer des politiques AWS gérées, y compris des fonctions de poste, à n'importe quelle identité IAM. Pour passer aux autorisations de moindre privilège, vous pouvez exécuter un analyseur Gestion des identités et des accès AWS d'accès pour surveiller les principaux utilisateurs à l'aide de politiques AWS gérées. Lorsque vous savez quelles autorisations ils utilisent, vous pouvez écrire une politique personnalisée ou générer une politique avec uniquement les autorisations requises pour votre équipe. Cette solution est moins sécurisée, mais offre plus de flexibilité à mesure que vous apprenez comment votre équipe l'utilise AWS.

AWS les politiques gérées pour les fonctions professionnelles sont conçues pour s'aligner étroitement sur les fonctions professionnelles courantes dans le secteur informatique. Vous pouvez utiliser ces politiques pour accorder les autorisations nécessaires afin d'exécuter les tâches prévues de la part quelqu'un occupant une fonction spécifique. Ces politiques regroupent les autorisations pour de nombreux services dans une seule politique plus facile à utiliser que des autorisations dispersées dans de nombreuses politiques.

Utiliser des rôles pour combiner les services

Certaines politiques utilisent des rôles de service IAM pour vous aider à tirer parti des fonctionnalités disponibles dans d'autres AWS services. Ces politiques accordent l'accès à`iam:passrole`, ce qui permet à un utilisateur disposant de la politique de transmettre un rôle à un AWS service. Ce rôle délègue les autorisations IAM au AWS service pour qu'il réalise des actions en votre nom.

Vous devez créer les rôles selon vos besoins. Par exemple, la politique d'administrateur réseau permet à un utilisateur doté de cette politique de transmettre un rôle nommé « flow-logs-vpc » au service Amazon. CloudWatch CloudWatch utilise ce rôle pour enregistrer et capturer le trafic IP des VPC créés par l'utilisateur.

Pour s'accorder aux meilleures pratiques de sécurité, les politiques pour les activités professionnelles incluent des filtres qui limitent les noms de rôles valides possibles à transmettre. Cela permet d'éviter d'accorder des autorisations inutiles. Si vos utilisateurs ont besoin des rôles de services facultatifs, vous devez créer un rôle qui suit la convention d'affectation de noms spécifiée dans la politique. Vous

pouvez ensuite accorder des autorisations pour le rôle. L'utilisateur peut alors configurer le service pour utiliser ce rôle, et lui octroyer toutes les autorisations fournies par le rôle.

Dans les sections suivantes, chaque nom de politique comporte un lien vers la page des détails de la politique dans Console de gestion AWS. Vous pouvez alors consulter le document de politique et examiner les autorisations qu'il accorde.

Fonction de tâche Administrateur

AWS nom de la politique gérée : AdministratorAccess

Cas d'utilisation : cet utilisateur a un accès total et peut déléguer des autorisations à tous les services et toutes les ressources dans AWS.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour de AWS politiques gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde toutes les actions pour tous les AWS services et pour toutes les ressources du compte. Pour plus d'informations sur la stratégie gérée, consultez le Guide AdministratorAccess de référence des politiques AWS gérées.

> **Note**
>
> Avant qu'un utilisateur ou un rôle IAM puisse accéder à la AWS Billing and Cost Management console avec les autorisations définies dans cette politique, vous devez d'abord activer l'accès aux utilisateurs et aux rôles IAM. Pour ce faire, suivez les instructions de la section Autoriser l'accès à la console de facturation pour déléguer l'accès à la console de facturation.

Fonction de tâche Facturation

AWS nom de la politique gérée : Facturation

Cas d'utilisation : cet utilisateur a besoin d'afficher les informations de facturation, de préparer les paiements et d'autoriser les paiements. L'utilisateur peut suivre les coûts cumulés pour l'ensemble du AWS service.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour de AWS politiques gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde la totalité des autorisations de gestion de la facturation, des coûts, des moyens de paiement et des rapports. Pour d'autres exemples de politique de gestion des coûts, consultez les exemples de politique AWS Billing dans le Guide de l'utilisateur AWS Billing and Cost Management Pour plus d'informations sur les politiques gérées, consultez Billing dans le Guide de référence des politiques gérées par AWS .

> **Note**
>
> Avant qu'un utilisateur ou un rôle IAM puisse accéder à la AWS Billing and Cost Management console avec les autorisations définies dans cette politique, vous devez d'abord activer l'accès aux utilisateurs et aux rôles IAM. Pour ce faire, suivez les instructions de la section Autoriser l'accès à la console de facturation pour déléguer l'accès à la console de facturation.

Fonction de tâche Administrateur de base de données

AWS nom de la politique gérée : DatabaseAdministrator

Cas d'utilisation : cet utilisateur configure et gère des bases de données dans le AWS Cloud.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour de AWS politiques gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde les autorisations de créer, configurer et gérer des bases de données. Il inclut l'accès à des services AWS de base de données, tels qu'Amazon DynamoDB, Amazon Relational Database Service (RDS) et Amazon Redshift. Consultez la politique pour la liste entière de services de base de données que prend en charge cette politique. Pour plus d'informations sur la stratégie gérée, consultez le Guide DatabaseAdministrator de référence des politiques AWS gérées.

Cette politique relative aux fonctions professionnelles favorise la possibilité de transférer des rôles aux AWS services. Elle autorise l'action `iam:PassRole` seulement pour les rôles nommés dans le tableau suivant. Pour plus d'informations, consultez Création des rôles et association des politiques (console) plus loin dans cette rubrique.

Cas d'utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélectionner	Sélectionnez cette politique AWS gérée
Permettre à l'utilisateur de surveiller des bases de données RDS	rds-monitoring-role	Rôle Amazon RDS pour la surveillance améliorée	AmazonRDS EnhancedMonitoring Role
Permettre AWS Lambda de surveiller votre base de données et d'accéder à des bases de données externes	rdbms-lambda-access	Amazon EC2	AWSLambda_FullAccess
Autoriser Lambda à télécharger des fichiers vers Amazon S3 et vers des clusters Amazon Redshift avec DynamoDB	lambda_exec_role	AWS Lambda	Créer une nouvelle politique gérée, tel que défini dans l'AWS Big Data Blog
Autoriser les fonctions Lambda à agir comme des déclencheurs pour vos tables DynamoDB	lambda-dynamodb-*	AWS Lambda	AWSLambda DynamoDBExecutionRole
Autoriser les fonctions Lambda à accéder à Amazon RDS dans un VPC	lambda-vpc-execution-role	Créer un rôle avec une politique d'approbation, tel que défini dans le Guide	AWSLambda VPCAccessExecution Role

Cas d'utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélectionner	Sélectionnez cette politique AWS gérée
		du développeur AWS Lambda	
AWS Data Pipeline Autoriser l'accès à vos AWS ressources	DataPipelineDefaultRole	Créer un rôle avec une politique d'approbation, tel que défini dans le Guide du développeur AWS Data Pipeline	La AWS Data Pipeline documentation répertorie les autorisations requises pour ce cas d'utilisation. Voir les rôles IAM pour AWS Data Pipeline
Permettre à vos applications qui s'exécutent sur des instances Amazon EC2 d'accéder à vos ressources AWS	DataPipelineDefaultResourceRole	Créer un rôle avec une politique d'approbation, tel que défini dans le Guide du développeur AWS Data Pipeline	AmazonEC2RoleforDataPipelineRole

Fonction de tâche Scientifique des données

AWS nom de la politique gérée : DataScientist

Cas d'utilisation : cet utilisateur exécute des tâches et des demandes Hadoop. L'utilisateur accède également à des informations pour l'analytique des données et la business intelligence, et analyse celles-ci.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les

mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour de AWS politiques gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde les autorisations nécessaires pour créer, gérer et exécuter des requêtes sur un cluster Amazon EMR et effectuer des analyses de données à l'aide d'outils tels qu'Amazon QuickSight. La politique inclut l'accès à des services de data scientists supplémentaires AWS Data Pipeline, tels qu'Amazon EC2, Amazon Kinesis, Amazon Machine Learning et SageMaker AI. Consultez la politique pour la liste entière de services de scientifique de données que prend en charge cette politique. Pour plus d'informations sur la stratégie gérée, consultez le Guide DataScientist de référence des politiques AWS gérées.

Cette politique relative aux fonctions professionnelles favorise la possibilité de transférer des rôles aux AWS services. Une déclaration permet de transmettre n'importe quel rôle à l' SageMaker IA. Une autre instruction autorise l'action `iam:PassRole` seulement pour les rôles nommés dans le tableau suivant. Pour plus d'informations, consultez Création des rôles et association des politiques (console) plus loin dans cette rubrique.

Cas d'utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélectionner	AWS politique gérée à sélectionner
Autoriser les instances Amazon EC2 à accéder aux services et aux ressources appropriés pour les clusters	EMR-EC2_DefaultRole	Amazon EMR pour EC2	AmazonElasticMapReduceforEC2Role
Autoriser Amazon EMR à accéder au service et aux ressources Amazon EC2 pour les clusters	EMR_ DefaultRole	Amazon EMR	AmazonEMRServicePolicy_v2
Autoriser le service géré Kinesis pour Apache Flink à accéder aux sources de données de diffusion	kinesis-*	Créer un rôle avec une politique d'approbation, tel que défini dans l'AWS Big Data Blog.	Consultez l'AWS Big Data Blog, qui définit les quatre options possibles en fonction de votre cas d'utilisation.

Cas d'utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélectionner	AWS politique gérée à sélectionner
AWS Data Pipeline Autoriser l'accès à vos AWS ressources	DataPipelineDefaultRole	Créer un rôle avec une politique d'approbation, tel que défini dans le Guide du développeur AWS Data Pipeline	La AWS Data Pipeline documentation répertorie les autorisations requises pour ce cas d'utilisation. Voir les rôles IAM pour AWS Data Pipeline
Permettre à vos applications qui s'exécutent sur des instances Amazon EC2 d'accéder à vos ressources AWS	DataPipelineDefaultResourceRole	Créer un rôle avec une politique d'approbation, tel que défini dans le Guide du développeur AWS Data Pipeline	AmazonEC2RoleforDataPipelineRole

Fonction de tâche Utilisateur avec pouvoir Développeur

AWS nom de la politique gérée : PowerUserAccess

Cas d'utilisation : cet utilisateur exécute des tâches de développement d'applications et peut créer et configurer des ressources et des services qui prennent en charge le développement d'applications AWS conscient.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour de AWS politiques gérées pour les fonctions professionnelles.

Description de la politique : La première déclaration de cette politique utilise l'`NotAction`élément pour autoriser toutes les actions pour tous les AWS services et pour toutes les ressources à l'exception de Gestion des identités et des accès AWS AWS Organizations, et Gestion de compte AWS. La seconde instruction accorde des autorisations IAM pour créer un rôle lié à un service. Elle est requise par certains services qui doivent accéder aux ressources d'un autre service, par exemple, un compartiment Amazon S3.

Il accorde également des AWS Organizations autorisations pour consulter les informations relatives à l'organisation de l'utilisateur, y compris l'adresse e-mail du compte de gestion et les limites de l'organisation. Bien que cette politique limite l'IAM AWS Organizations, elle permet à l'utilisateur d'effectuer toutes les actions IAM Identity Center si IAM Identity Center est activé. Il accorde également des autorisations à la gestion du compte pour voir quelles AWS régions sont activées ou désactivées pour le compte.

Fonction de tâche Administrateur réseau

AWS nom de la politique gérée : NetworkAdministrator

Cas d'utilisation : cet utilisateur est chargé de configurer et de gérer les ressources AWS réseau.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour de AWS politiques gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde des autorisations pour créer et gérer des ressources réseau dans Auto Scaling, Amazon EC2 AWS Direct Connect, Route 53, Amazon CloudFront, Elastic Load Balancing AWS Elastic Beanstalk, Amazon SNS, CloudWatch Logs CloudWatch, Amazon S3, IAM et Amazon Virtual Private Cloud. Pour plus d'informations sur la stratégie gérée, consultez le Guide NetworkAdministrator de référence des politiques AWS gérées.

Cette fonction nécessite la capacité de transférer des rôles aux AWS services. Elle accorde `iam:GetRole` et `iam:PassRole` seulement pour les rôles nommés dans le tableau suivant. Pour plus d'informations, consultez Création des rôles et association des politiques (console) plus loin dans cette rubrique.

Cas d'utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélectionner	AWS politique gérée à sélectionner
Permet à Amazon VPC de créer et de gérer les journaux et CloudWatch journaux pour le compte de l'utilisateur afin de surveiller le trafic IP entrant et sortant de votre VPC	flow-logs-*	Créer un rôle avec une politique d'approbation, tel que défini dans le Guide de l'utilisateur Amazon VPC	Ce cas d'utilisation ne dispose pas d'une politique AWS gérée existante, mais la documentation répertorie les autorisations requises. Consultez le Guide de l'utilisateur Amazon VPC.

Read-only accès

AWS nom de la politique gérée : ReadOnlyAccess

Cas d'utilisation : cet utilisateur nécessite un accès en lecture seule à toutes les ressources d'un Compte AWS.

> **Important**
>
> Cet utilisateur aura également accès à la lecture des données dans des services de stockage tels que les compartiments Amazon S3 et les tables Amazon DynamoDB.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour de AWS politiques gérées pour les fonctions professionnelles.

Description de la politique : cette politique octroie les autorisations de répertorier, obtenir, décrire ou afficher les ressources et leurs attributs. Elle n'inclut pas des fonctions de mutation, telles que créer ou supprimer. Cette politique inclut l'accès en lecture seule aux AWS services liés à la sécurité, tels que et. Gestion des identités et des accès AWS AWS Billing and Cost Management

Consultez la politique pour la liste entière des services et actions que prend en charge cette politique. Pour plus d'informations sur la stratégie gérée, consultez le Guide ReadOnlyAccess de référence des politiques AWS gérées. Si vous avez besoin d'une politique similaire qui n'accorde pas l'accès à la lecture des données dans les services de stockage, consultez View-only fonction de travail utilisateur.

Accès complet aux actions du service MCP

AWS nom de la politique gérée : AWSMcpServiceActionsFullAccess

Cas d'utilisation : cet utilisateur a besoin d'accéder à AWS des services via des serveurs AWS MCP. Cette politique n'accorde pas l'accès aux actions entreprises par un service MCP à d'autres AWS services.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour de AWS politiques gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde l'autorisation d'appeler n'importe quelle action de service AWS MCP. Vous pouvez l'utiliser lorsque vous n'avez pas besoin de spécifier d'autorisations par service AWS MCP. Il n'accorde pas d'autorisations aux actions entreprises par le service MCP à d'autres AWS services. Ces autorisations doivent toujours être accordées séparément et en plus des actions du service MCP. Pour plus d'informations sur la stratégie gérée, consultez le Guide AWSMcpServiceActionsFullAccess de référence des politiques AWS gérées.

Fonction de tâche Audit de sécurité

AWS nom de la politique gérée : SecurityAudit

Cas d'utilisation : cet utilisateur surveille les comptes pour vérifier leur conformité aux exigences de sécurité. Il peut accéder aux journaux et aux événements pour rechercher des brèches de sécurité potentielles ou d'éventuelles activités malveillantes.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour de AWS politiques gérées pour les fonctions professionnelles.

Description de la politique : cette politique accorde l'autorisation d'afficher les données de configuration de nombreux AWS services et de consulter leurs journaux. Pour plus d'informations sur la stratégie gérée, consultez le Guide SecurityAudit de référence des politiques AWS gérées.

Fonction de tâche Utilisateur support

AWS nom de la politique gérée : AWSSupportAccess

Cas d'utilisation : cet utilisateur contacte le AWS support, crée des demandes d'assistance et consulte l'état des demandes existantes.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour de AWS politiques gérées pour les fonctions professionnelles.

Description de la politique : cette politique accorde des autorisations pour créer et mettre à jour Support des dossiers. Pour plus d'informations sur la stratégie gérée, consultez le Guide AWSSupportAccess de référence des politiques AWS gérées.

Fonction de tâche Administrateur système

AWS nom de la politique gérée : SystemAdministrator

Cas d'utilisation : cet utilisateur configure et gère les ressources pour les opérations de développement.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour de AWS politiques gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde des autorisations pour créer et gérer des ressources pour une grande variété de AWS services AWS CloudTrail, notamment Amazon CloudWatch AWS CodeCommit, AWS CodeDeploy, AWS Config, AWS Directory Service,, Amazon EC2 Gestion des identités et des accès AWS, AWS Key Management Service, AWS Lambda, Amazon RDS, Route 53, Amazon S3, Amazon SES, Amazon SQS et Amazon AWS Trusted Advisor VPC. Pour plus d'informations sur la stratégie gérée, consultez le Guide SystemAdministrator de référence des politiques AWS gérées.

Cette fonction nécessite la capacité de transférer des rôles aux AWS services. Elle accorde `iam:GetRole` et `iam:PassRole` seulement pour les rôles nommés dans le tableau suivant. Pour plus d'informations, consultez Création des rôles et association des politiques (console) plus loin dans cette rubrique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour de AWS politiques gérées pour les fonctions professionnelles.

Cas d'utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélectionner	AWS politique gérée à sélectionner
Autoriser les applications s'exécutant dans des instances EC2 d'un cluster Amazon ECS à accéder à Amazon ECS	ecr-sysadmin-*	Rôle Amazon EC2 pour EC2 Container Service	AmazonEC2ContainerServiceforEC2Role
Permettre à un utilisateur de surveiller des bases de données	rds-monitoring-role	Rôle Amazon RDS pour la surveillance améliorée	AmazonRDSEnhancedMonitoringRole
Autorisez les applications exécutées dans les instances EC2 à accéder aux AWS ressources.	ec2-sysadmin-*	Amazon EC2	Exemple de politique de rôle qui accorde l'accès à un compartiment S3, comme illustré dans le Guide de l'utilisateur Amazon EC2 ; à personnal

Cas d'utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélectionner	AWS politique gérée à sélectionner
			iser en fonction des besoins
Autoriser Lambda à lire les flux DynamoDB et à écrire dans les journaux CloudWatch	lambda-sysadmin-*	AWS Lambda	AWSLambdaDynamoDBExecutionRole

View-only fonction de travail utilisateur

AWS nom de la politique gérée : ViewOnlyAccess

Cas d'utilisation : cet utilisateur peut consulter une liste des AWS ressources et des métadonnées de base du compte pour tous les services. Il ne peut pas lire le contenu des ressources ou les metadonnées au-delà des informations de quotas et de listes des ressources.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour de AWS politiques gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde `List*Describe*`,, `Get*View*`, et `Lookup*` l'accès aux ressources pour les AWS services. Pour savoir quelles actions cette politique inclut pour chaque service, consultez ViewOnlyAccess. Pour plus d'informations sur la stratégie gérée, consultez le Guide ViewOnlyAccess de référence des politiques AWS gérées.

Mises à jour de AWS politiques gérées pour les fonctions professionnelles

Ces politiques sont toutes maintenues AWS et mises à jour pour inclure la prise en charge des nouveaux services et des nouvelles fonctionnalités au fur et à mesure qu'ils sont ajoutés par les AWS services. Elles ne peuvent pas être modifiées par les clients. Vous pouvez créer une copie de la politique, puis la modifier, mais cette copie n'est pas automatiquement mise à jour en raison de l' AWS introduction de nouveaux services et opérations d'API.

Pour une politique de fonction de tâche, vous pouvez afficher l'historique des versions, ainsi que l'heure et la date de chaque mise à jour dans la console IAM. Pour cela, utilisez les liens sur cette page pour afficher les détails de la politique. Ensuite, choisissez l'onglet Policy versions (Versions de politique) pour afficher les versions. Cette page affiche les 25 dernières versions d'une politique. Pour afficher toutes les versions d'une politique, appelez la AWS CLI commande get-policy-version ou l'GetPolicyVersionopération API.

> **Note**
>
> Vous pouvez avoir jusqu'à cinq versions d'une politique gérée par le client, tout en AWS conservant l'historique complet des versions des politiques AWS gérées.

Création des rôles et association des politiques (console)

Plusieurs des politiques répertoriées précédemment permettent de configurer des AWS services avec des rôles qui permettent à ces services d'effectuer des opérations en votre nom. Les politiques de fonctions professionnelles spécifient les noms de rôles exacts à utiliser ou incluent au moins un préfixe qui indique la première partie du nom qui peut être utilisé. Pour créer un de ces rôles, suivez les étapes de la procédure ci-dessous.

Pour créer un rôle pour Service AWS (console IAM)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le volet de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
3. Pour Trusted entity (Entité de confiance), choisissez Service AWS.
4. Pour Service ou cas d'utilisation, choisissez un service, puis choisissez le cas d'utilisation. Les cas d'utilisation sont définis par le service pour inclure la politique d'approbation nécessaire au service.
5. Choisissez Suivant.
6. Pour Politiques d'autorisations, les options dépendent du cas d'utilisation que vous avez sélectionné :

   - Si le service définit les autorisations pour le rôle, il n'est pas possible de sélectionner les politiques d'autorisation.

- Choisissez parmi un ensemble limité de politiques d'autorisation.
- Choisissez parmi toutes les politiques d'autorisation.
- Ne sélectionnez aucune politique d'autorisation, créez les politiques une fois le rôle créé, puis attachez-les au rôle.

7. (Facultatif) Définissez une limite d'autorisations. Il s'agit d'une fonctionnalité avancée disponible pour les fonctions de service, mais pas pour les rôles liés à un service.

   a. Ouvrez la section Définir une limite des autorisations et choisissez Utiliser une limite des autorisations pour contrôler le nombre maximum d'autorisations de rôle.

      IAM inclut une liste des politiques AWS gérées et gérées par le client dans votre compte.

   b. Sélectionnez la politique à utiliser comme limite d'autorisations.

8. Choisissez Suivant.
9. Pour Nom du rôle, les options dépendent du service :

   - Si le service définit le nom du rôle, vous ne pouvez pas modifier le nom du rôle.
   - Si le service définit un préfixe pour le nom du rôle, vous pouvez saisir un suffixe facultatif.
   - Si le service ne définit pas le nom du rôle, vous pouvez le nommer.

   > **Important**
   >
   > Lorsque vous nommez un rôle, notez ce qui suit :
   >
   > - Les noms de rôles doivent être uniques au sein de votre Compte AWS organisation et ne peuvent pas être rendus uniques au cas par cas.
   >
   >   Par exemple, ne créez pas deux rôles nommés **`PRODROLE`** et **`prodrole`**. Lorsqu'un nom de rôle est utilisé dans une politique ou dans le cadre d'un ARN, le nom de rôle est sensible à la casse. Cependant, lorsqu'un nom de rôle apparaît aux clients dans la console, par exemple lors de la procédure d'ouverture de session, le nom de rôle est insensible à la casse.
   > - Vous ne pouvez pas modifier le nom du rôle après sa création, car d'autres entités pourraient y faire référence.

10. (Facultatif) Pour Description, saisissez la description du rôle.

11. (Facultatif) Pour modifier les cas d'utilisation et les autorisations du rôle, dans les sections Étape 1 : sélectionner les entités de confiance ou Étape 2 : ajouter des autorisations, sélectionnez Modifier.
12. (Facultatif) Pour identifier, organiser ou rechercher le rôle, ajoutez des identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, consultez la section Balises pour les Gestion des identités et des accès AWS ressources dans le Guide de l'utilisateur IAM.
13. Passez en revue les informations du rôle, puis choisissez Create role (Créer un rôle).

Exemple 1 : Configuration d'un utilisateur en tant qu'administrateur de base de données (console)

Cet exemple illustre les étapes nécessaires pour configurer Alice, un utilisateur IAM, en tant que Database Administrator (Administrateur de base de données). Vous utilisez les informations dans la première ligne de la table de cette section et autorisez l'utilisateur à activer la surveillance d'Amazon RDS. Vous associez la DatabaseAdministrator politique à l'utilisateur IAM d'Alice afin qu'il puisse gérer les services de base de données Amazon. Cette politique permet également à Alice de transmettre un rôle appelé `rds-monitoring-role` au service Amazon RDS qui autorise ce dernier à superviser les bases de données Amazon RDS en son nom.

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Choisissez Politiques, tapez **`database`** dans la zone de recherche, puis appuyez sur Entrée.
3. Sélectionnez le bouton radio correspondant à la DatabaseAdministrator politique, choisissez Actions, puis choisissez Joindre.
4. Dans la liste des utilisateurs, sélectionnez Alice, puis choisissez Attacher la politique. Alice peut désormais administrer des AWS bases de données. Cependant, pour permettre à Alice de surveiller ces bases de données, vous devez configurer le rôle du service.
5. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
6. Choisissez le type de fonction du AWS service (Service), puis Amazon RDS.
7. Sélectionnez le cas d'utilisation Amazon RDS Role for Enhanced Monitoring.
8. Amazon RDS définit les autorisations pour votre rôle. Choisissez Suivant : Vérification pour continuer.

9. Le nom du rôle doit être l'un de ceux spécifiés par la DatabaseAdministrator politique actuelle d'Alice. L'un de ces noms est **`rds-monitoring-role`**. Saisissez ceci pour le Role name (Nom du rôle).
10. (Facultatif) Dans le champ Description du rôle, saisissez la description du nouveau rôle.
11. Après avoir passé en revue les détails, choisissez Créer un rôle.
12. Alice peut désormais activer RDS Enhanced Monitoring (Surveillance améliorée RDS) dans la section Monitoring (Surveillance) de la console Amazon RDS. Par exemple, elle peut utiliser cette fonction lorsqu'elle crée une instance de base de données ou qu'elle crée un réplica en lecture, ou quand elle modifie une instance de base de données. Elle doit saisir le nom de rôle qu'elle a créé (rds-monitoring-role) dans la boîte Monitoring Role (Rôle de surveillance) lorsqu'elle définit Enable Enhanced Monitoring (Activer la surveillance améliorée) sur Yes (Oui).

Exemple 2 : Configuration d'un utilisateur en tant qu'administrateur réseau (console)

Cet exemple montre les étapes nécessaires pour configurer Juan, un utilisateur IAM, en tant que Administrateur réseau. Il utilise les informations de la table de cette section pour autoriser Juan à superviser le trafic IP entrant et sortant d'un VPC. Cela permet également à Jorge de capturer ces informations dans les journaux de CloudWatch Logs. Vous associez la NetworkAdministrator politique à l'utilisateur IAM de Jorge afin qu'il puisse configurer les ressources AWS réseau. Cette politique permet également à Juan de transmettre un rôle dont le nom commence par `flow-logs*` à Amazon EC2 lorsque vous créez un journal de flux. Dans ce scénario, à la différence de l'exemple 1, il n'y a aucun type de rôle de service prédéfini. Vous devez donc procéder différemment pour certaines étapes.

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans le panneau de navigation, choisissez Politiques, puis saisissez **`network`** dans la zone de recherche et appuyez sur Entrée.
3. Sélectionnez le bouton radio à côté de NetworkAdministrator la politique, choisissez Actions, puis choisissez Joindre.
4. Dans la liste des utilisateurs, cochez la case en regard de Juan, puis choisissez Attacher la politique. Jorge peut désormais administrer les ressources AWS du réseau. Cependant, pour activer la surveillance du trafic IP dans votre VPC, vous devez configurer le rôle de service.
5. Comme le rôle de service à créer n'a pas de politique gérée prédéfinie, il doit d'abord être créé. Dans le panneau de navigation, sélectionnez Politicies (Politiques), puis Create policy (Créer une politique).

6. Dans la section Éditeur de politiques, choisissez l'option JSON et copiez le texte du document de stratégie JSON suivant. Collez ce texte dans la zone de texte JSON.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

7. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant.

   > **Note**
   >
   > Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour de plus amples informations, veuillez consulter Restructuration de politique.

8. Sur la page Vérifier et créer, tapez **`vpc-flow-logs-policy-for-service-role`** pour le nom de la politique. Vérifiez les Autorisations définies dans cette politique pour voir les autorisations accordées par votre politique, puis choisissez Créer une politique pour enregistrer votre travail.

   La nouvelle politique s'affiche dans la liste des politiques gérées et est prête à être attachée.

9. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).

10. Choisissez le type de fonction du AWS service (Service), puis Amazon EC2.
11. Sélectionnez le cas d'utilisation Amazon EC2.
12. Sur la page Attach permissions policies (Attacher des politiques d'autorisations), choisissez la politique que vous avez créée, vpc-flow-logs-policy-for-service-role, puis Suivant : Vérification.
13. Le nom du rôle doit être autorisé par la NetworkAdministrator politique actuelle de Jorge. Tout nom qui commence par `flow-logs-` est autorisé. Pour cet exemple, saisissez **`flow-logs-for-jorge`** comme Role name (Nom du rôle).
14. (Facultatif) Dans le champ Description du rôle, saisissez la description du nouveau rôle.
15. Après avoir passé en revue les détails, choisissez Créer un rôle.
16. Vous pouvez désormais configurer la politique d'approbation nécessaire pour ce scénario. Dans la page Rôles, choisissez le rôle flow-logs-for-jorge (journaux-de-flux-pour-juan), le nom, pas la case à cocher. Sur la page des détails de votre nouveau rôle, choisissez l'onglet Relations d'approbation, puis choisissez Modifier la relation d'approbation.
17. Modifier la ligne « Service » comme suit, en remplaçant l'entrée pour `ec2.amazonaws.com` :

```
        "Service": "vpc-flow-logs.amazonaws.com"
```

18. Juan peut désormais créer des journaux de flux pour un VPC ou un sous-réseau dans la console Amazon EC2. Lorsque vous créez le journal de flux, spécifiez le rôle flow-logs-for-jorge (journaux-de-flux-pour-juan). Ce rôle dispose des autorisations pour créer le journal et y consigner des données.

AWS clés de contexte de condition globale

Lorsqu'un responsable fait une demande à AWS, AWS rassemble les informations de demande dans un contexte de demande. Vous pouvez utiliser l'élément `Condition` d'une politique JSON pour comparer des clés dans le contexte de demande avec les valeurs de clé spécifiées dans votre politique. Les informations sur les requêtes proviennent de différentes sources, notamment du principal qui fait la requête, de la ressource sur laquelle porte la requête et des métadonnées relatives à la requête elle-même.

Les clés de condition globales peuvent être utilisées dans tous les services AWS . Bien que ces clés de condition puissent être utilisées dans toutes les politiques, elles ne sont pas disponibles dans tous les contextes de requête. Par exemple, la clé de condition `aws:SourceAccount` n'est disponible que lorsque l'appel à votre ressource est effectué directement par un principal de service AWS.

Pour savoir quand une clé globale figure dans le contexte de la requête, consultez les informations Disponibilité pour chaque clé.

Certains services individuels créent leurs propres clés de condition qui sont disponibles dans le contexte de la demande pour d'autres services. Cross-service les clés de condition sont un type de clé de condition globale qui inclut un préfixe correspondant au nom du service, tel que `ec2:` ou `lambda:`, mais qui sont disponibles pour d'autres services.

Service-specific les clés de condition sont définies pour être utilisées avec un AWS service individuel. Par exemple, Amazon S3 vous permet de rédiger une politique avec la clé de condition `s3:VersionId` pour limiter l'accès à une version spécifique d'un objet Amazon S3. Cette clé de condition est propre au service, ce qui signifie qu'elle ne fonctionne qu'avec les requêtes adressées au service Amazon S3. Pour les clés de condition spécifiques à un service, voir Actions, ressources et clés de condition pour les AWS services et choisissez le service dont vous souhaitez afficher les clés.

> **Note**
>
> Si vous utilisez des clés de condition qui ne sont disponibles que dans certaines circonstances, vous pouvez utiliser les IfExists versions des opérateurs de condition. Si les clés de condition ne figurent pas dans le contexte de la demande, l'évaluation de la politique peut échouer. Par exemple, utilisez le bloc de condition suivant avec les opérateurs `...IfExists` pour déterminer quand la demande provient d'une plage d'adresses IP ou d'un VPC spécifique. Si l'une des clés, ou les deux, ne figurent pas dans le contexte de la demande, la condition renvoie toujours `true`. Les valeurs ne sont vérifiées que si la clé spécifiée figure dans le contexte de la demande. Pour plus d'informations sur la manière dont une politique est évaluée lorsqu'une clé n'est pas présente pour d'autres opérateurs, consultez la section Opérateurs de condition.
>
> ```
> "Condition": {
>     "IpAddressIfExists": {"aws:SourceIp" : ["xxx"] },
>     "StringEqualsIfExists" : {"aws:SourceVpc" : ["yyy"]}
> }
> ```

> **Important**
>
> Pour comparer votre condition à un contexte de demande avec plusieurs valeurs de clé, vous devez utiliser les opérateurs d'ensemble `ForAllValues` ou `ForAnyValue`. Utilisez des opérateurs d'ensemble uniquement avec des clés de condition à valeurs multiples. N'utilisez pas d'opérateurs d'ensemble avec des clés de condition à valeur unique. Pour de plus amples informations, veuillez consulter Définir les opérateurs pour les clés de contexte à valeurs multiples.

Propriétés du principal	Propriétés d'une session de rôle	Propriétés du réseau	Propriétés de la ressource	Propriétés de la requête
aws : PrincipalArn aws : PrincipalAccount aws : PrincipalOrgPaths aws : PrincipalOrg ID les lois :PrincipalTag/ *clé-balise* aws : PrincipalIsAWSService	aws : AssumedRoot aws : FederatedProvider aws : TokenIssueTime aws : SignInSessionArn aws : MultiFactorAuthAge aws : MultiFactorAuthPresent	aws : SourceIp aws : SourceVpc aws : SourceVpcArn aws : SourceVpce aws : VpceAccount aws : VpceOrg ID aws : VpceOrgPaths aws : VpcSourceIp	aws : ResourceAccount aws : ResourceOrg ID aws : ResourceOrgPaths les lois :ResourceTag/ *clé-balise*	aws : CalledVia aws : CalledViaFirst aws : CalledViaLast en tant que : CalledViaAWSMCP aws:ViaAWSService AWS : via le service AWS SMCP aws : CurrentTime

Propriétés du principal	Propriétés d'une session de rôle	Propriétés du réseau	Propriétés de la ressource	Propriétés de la requête
aws : PrincipalServiceName aws : PrincipalServiceNamesList aws : PrincipalType aws:userid aws:username	aws : ChatbotSourceArn aws : Ec2InstanceSourceVpc aws : Ec2InstanceSourcePrivateIPv4 aws : SourceIdentity aws : ViaCustomerDomain EC2 : RoleDelivery EC2 : SourceInstanceArn colle : RoleAssumedBy colle : Credentia			aws : EpochTime aws:referer aws : RequestedRegion les lois :RequestTag/ *clé-balise* aws : TagKeys aws : SecureTransport aws : SourceAccount aws : SourceArn aws : SourceOrg ID aws : SourceOrg Paths

Propriétés du principal	Propriétés d'une session de rôle	Propriétés du réseau	Propriétés de la ressource	Propriétés de la requête
	lIssuingService construction de code : BuildArn construction de code : ProjectArn lambda : SourceFunctionArn sms : SourceInstanceArn boutique d'identité : UserId			aws : UserAgent aws : IsMcpServiceAction

Clés de condition sensibles

Les clés de condition suivantes sont considérées comme sensibles. Le recours à des caractères génériques dans ces clés de condition ne répond à aucun cas d'utilisation valable, même avec une sous-chaîne de la valeur de la clé contenant un caractère générique. En effet, le caractère générique peut faire correspondre la clé de condition à n'importe quelle valeur, ce qui peut présenter un risque de sécurité.

- aws : PrincipalAccount
- aws : PrincipalOrg ID

- aws : ResourceAccount
- aws : ResourceOrg ID
- aws : SourceAccount
- aws : SourceOrg ID
- aws : SourceVpc
- aws : SourceVpce
- aws : VpceAccount
- aws : VpceOrg ID

Propriétés du principal

Utilisez les clés de condition suivantes pour comparer les détails concernant le principal qui effectue la requête avec les propriétés du principal que vous spécifiez dans la politique. Pour obtenir la liste des principaux qui peuvent effectuer des requêtes, consultez Comment spécifier un principal.

aws : PrincipalArn

Utilisez cette clé pour comparer l'Amazon Resource Name (ARN) du principal ayant fait la demande avec l'ARN spécifié dans la politique. Pour les rôles IAM, le contexte de la demande renvoie l'ARN du rôle, et non l'ARN de l'utilisateur qui a endossé le rôle.

- Availability (Disponibilité) : cette clé figure dans le contexte de la demande pour toutes les demandes signées. Les demandes anonymes n'incluent pas cette clé. Vous pouvez spécifier les types de mandataires suivants dans cette clé de condition :
  - Rôle IAM
  - Utilisateur IAM
  - AWS STS utilisateur principal fédéré
  - Compte AWS utilisateur root
- Type de données : ARN

  AWS recommande d'utiliser des opérateurs ARN plutôt que des opérateurs de chaîne lors de la comparaison des ARN.
- Type de valeur — Single-valued

- Valeurs d'exemple La liste suivante indique la valeur de contexte de requête renvoyée pour les différents types de principaux que vous pouvez spécifier dans la clé de condition `aws:PrincipalArn` :
  - Rôle IAM— Le contexte de la requête contient la valeur suivante pour la clé de condition `aws:PrincipalArn`. Ne spécifiez pas l'ARN de session de rôle présumé comme valeur pour cette clé de condition. Pour en savoir plus sur le rôle présumé de session principale, reportez-vous à la section Principaux de séance de rôle.

    ```
    arn:aws:iam::123456789012:role/role-name
    ```

  - Utilisateur IAM— Le contexte de la requête contient la valeur suivante pour la clé de conditionaws:PrincipalArn.

    ```
    arn:aws:iam::123456789012:user/user-name
    ```

  - AWS STS principes d'utilisateur fédérés : le contexte de la demande contient la valeur suivante pour la clé de condition. `aws:PrincipalArn`

    ```
    arn:aws:sts::123456789012:federated-user/user-name
    ```

  - Compte AWS utilisateur root : le contexte de la demande contient la valeur suivante pour la clé de conditionaws:PrincipalArn. Lorsque vous spécifiez l'ARN de l'utilisateur root comme valeur pour la `aws:PrincipalArn` clé de condition, il ne limite les autorisations que pour l'utilisateur root de Compte AWS. Cela diffère de la spécification de l'ARN de l'utilisateur root dans l'élément principal d'une politique basée sur les ressources, qui délègue l'autorité au Compte AWS. Pour en savoir plus sur la spécification de l'ARN de l'utilisateur root dans l'élément principal d'une politique basée sur les ressources, consultez Compte AWS principals.

    ```
    arn:aws:iam::123456789012:root
    ```

Vous pouvez spécifier l'ARN de l'utilisateur root en tant que valeur pour la clé de condition `aws:PrincipalArn` dans les politiques de contrôle des AWS Organizations services (SCP). Les SCP sont un type de stratégie d'organisation que vous pouvez utiliser pour gérer les autorisations dans votre organisation ; elles n'affectent que les comptes des membres dans l'organisation. Une SCP limite les autorisations des utilisateurs et des rôles IAM dans les comptes membres, y compris l'utilisateur racine du compte membre. Pour plus d'informations sur l'effet des SCP sur

les autorisations, consultez les effets des SCP sur les autorisations dans le Guide de l'utilisateur AWS Organizations .

aws : PrincipalAccount

Utilisez cette clé pour comparer le compte auquel appartient le principal demandeur avec l'identifiant de compte spécifié dans la politique. Pour les demandes anonymes, le contexte de la demande renvoie `anonymous`.

- Disponibilité : cette clé figure dans le contexte de la demande pour toutes les demandes, y compris les requêtes anonymes.
- Types de données – Chaîne
- Type de valeur — Single-valued

Dans l'exemple suivant, l'accès est refusé, sauf aux principaux possédant le numéro de compte `123456789012`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAccessFromPrincipalNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:us-east-1:111122223333:resource"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:PrincipalAccount": [
            "123456789012"
          ]
        }
      }
    }
  ]
}
```

aws : PrincipalOrgPaths

Utilisez cette clé pour comparer le AWS Organizations chemin du principal qui fait la demande au chemin de la politique. Ce principal peut être un utilisateur IAM, un rôle IAM, un principal utilisateur fédéré AWS STS ou un Utilisateur racine d'un compte AWS. Dans une politique, cette clé de condition vérifie que le demandeur est un membre du compte au sein de l'organisation racine ou des unités d'organisation spécifiées dans AWS Organizations. Un AWS Organizations chemin est une représentation textuelle de la structure d'une AWS Organizations entité. Pour de plus amples informations sur l'utilisation et la compréhension des chemins, veuillez consulter Comprenez les AWS Organizations chemin de l'entité.

- Availability (Disponibilité) : cette clé ne figure dans le contexte de la demande que si le principal est membre d'une organisation. Les demandes anonymes n'incluent pas cette clé.
- Type de données – Chaîne (liste)
- Type de valeur – À valeur multiple

> **Note**
>
> Les ID d'organisation sont globalement uniques, mais les ID d'unité d'organisation et les ID racine ne sont uniques qu'au sein d'une organisation. Cela signifie qu'aucune organisation ne partage le même ID d'organisation. Toutefois, une autre organisation peut avoir le même ID d'unité d'organisation ou ID racine que vous. Nous vous recommandons de toujours inclure l'ID d'organisation lorsque vous spécifiez une unité d'organisation ou une racine.

Par exemple, la condition suivante renvoie `true` pour les principaux de comptes qui sont directement attachés à l'unité d'organisation `ou-ab12-22222222`, et non à ses unités d'organisation enfants.

```
"Condition" : { "ForAnyValue:StringEquals" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
}}
```

La condition suivante renvoie `true` pour les principaux d'un compte qui est directement attaché à l'unité d'organisation ou à l'une de ses unités d'organisation enfants. Lorsque vous incluez un caractère générique, vous devez utiliser l'opérateur de condition `StringLike`.

```
"Condition" : { "ForAnyValue:StringLike" : {
```

```
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
*"]
}}
```

La condition suivante renvoie `true` pour les principaux d'un compte qui est directement attaché à l'une des unités d'organisation enfant, mais pas directement à l'unité d'organisation parent. La condition précédente s'applique à l'unité d'organisation ou à tous les enfants. La condition suivante s'applique uniquement aux enfants (et aux enfants de ces enfants).

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
ou-*"]
}}
```

La condition suivante autorise l'accès à chaque principal de l'organisation `o-a1b2c3d4e5`, quelle que soit l'unité d'organisation parent.

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/*"]
}}
```

`aws:PrincipalOrgPaths` est une clé de condition à valeurs multiples. Les clés multivaluées peuvent avoir plusieurs valeurs dans le contexte de la demande. Lorsque vous utilisez plusieurs valeurs avec l'opérateur de condition `ForAnyValue`, le chemin d'accès du principal doit correspondre à l'un des chemins spécifiés dans la politique. Pour de plus amples informations sur les clés de condition à valeurs multiples, veuillez consulter Définir les opérateurs pour les clés de contexte à valeurs multiples.

```
    "Condition": {
        "ForAnyValue:StringLike": {
            "aws:PrincipalOrgPaths": [
                "o-a1b2c3d4e5/r-ab12/ou-ab12-33333333/*",
                "o-a1b2c3d4e5/r-ab12/ou-ab12-22222222/*"
            ]
        }
    }
```

aws : PrincipalOrg ID

Utilisez cette clé pour comparer l'identifiant de l'organisation AWS Organizations à laquelle appartient le principal demandeur avec l'identifiant spécifié dans la politique.

- Availability (Disponibilité) : cette clé ne figure dans le contexte de la demande que si le principal est membre d'une organisation. Les demandes anonymes n'incluent pas cette clé.
- Types de données – Chaîne
- Type de valeur — Single-valued

Cette clé globale permet d'éviter de répertorier tous les ID de compte pour tous les comptes AWS d'une organisation. Vous pouvez utiliser cette clé de condition pour simplifier la spécification de l'élément `Principal` d'une politique basée sur une ressource. Vous pouvez spécifier l'ID de l'organisation dans l'élément Condition. Lorsque vous ajoutez et supprimez des comptes, les politiques qui contiennent la clé `aws:PrincipalOrgID` ajoutent automatiquement les bons comptes et ne nécessitent pas de mise à jour manuelle.

Par exemple, la politique de compartiment Amazon S3 suivante permet aux membres de n'importe quel compte de l'organisation `o-xxxxxxxxxxx` d'ajouter un objet dans le compartiment `amzn-s3-demo-bucket`.

JSON

```
{
 "Version":"2012-10-17",
 "Statement": {
   "Sid": "AllowPutObject",
   "Effect": "Allow",
   "Principal": "*",
   "Action": "s3:PutObject",
   "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
   "Condition": {"StringEquals":
     {"aws:PrincipalOrgID":"o-xxxxxxxxxxx"}
   }
 }
}
```

> **Note**
>
> Cette condition globale s'applique également au compte de gestion d'une organisation AWS . Cette politique empêche tous les principaux en dehors de l'organisation spécifiée d'accéder au compartiment Amazon S3. Cela inclut tous AWS les services qui interagissent avec vos ressources internes, tels que AWS CloudTrail l'envoi de données de journal à vos compartiments Amazon S3. Pour savoir comment autoriser l'accès aux AWS services en toute sécurité, consultezaws : PrincipalIsAWSService.

Pour plus d'informations sur AWS Organizations, voir Qu'est-ce que AWS Organizations ? dans le Guide de AWS Organizations l'utilisateur.

les lois :PrincipalTag/clé-balise

Utilisez cette clé pour comparer la balise attachée au principal effectuant la demande avec la balise spécifiée dans la politique. Si plusieurs balises sont attachées au principal, le contexte de la demande inclut une clé `aws:PrincipalTag` pour chaque clé de balise attachée.

- Availability (Disponibilité) : cette clé figure dans le contexte de la demande si le principal utilise un utilisateur IAM avec des balises attachées. Elle est incluse pour un principal utilisant un rôle IAM avec des balises attachées ou des balises de session. Les demandes anonymes n'incluent pas cette clé.
- Types de données – Chaîne
- Type de valeur — Single-valued

Vous pouvez ajouter des attributs personnalisés à un utilisateur ou un rôle sous la forme d'une paire clé-valeur. Pour plus d'informations sur les balises IAM, consultez Tags pour Gestion des identités et des accès AWS resources. Vous pouvez utiliser `aws:PrincipalTag`pour le contrôle d'accès des principaux AWS .

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise les utilisateurs avec la balise **`department=hr`** à gérer les utilisateurs, groupes ou rôles IAM. Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:Get*",
        "iam:List*",
        "iam:Generate*"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": "hr"
        }
      }
    }
  ]
}
```

aws : PrincipalIsAWSService

Utilisez cette touche pour vérifier si l'appel à votre ressource est effectué directement par un responsable AWS de service. Par exemple, AWS CloudTrail utilise le principal de service `cloudtrail.amazonaws.com` pour écrire des journaux dans votre compartiment Amazon S3. La clé de contexte de demande est définie sur true lorsqu'un service utilise un principal de service pour effectuer une action directe sur vos ressources. La clé de contexte est définie sur false si le service utilise les informations d'identification d'un principal IAM pour effectuer une demande au nom du principal. Elle est également définie sur false si le service utilise un rôle de service ou un rôle lié à un service pour effectuer un appel au nom du principal.

- Disponibilité : cette clé est présente dans le contexte de la demande pour toutes les demandes d'API signées utilisant des informations d'identification AWS . Les demandes anonymes n'incluent pas cette clé.
- Types de données – Booléen
- Type de valeur — Single-valued

Vous pouvez utiliser cette clé de condition pour limiter l'accès à vos identités fiables et aux emplacements réseau attendus tout en accordant l'accès aux AWS services en toute sécurité.

Dans l'exemple de politique de compartiment Amazon S3 suivant, l'accès au compartiment est restreint, sauf si la demande provient `vpc-111bbb22` ou provient d'un principal de service, tel que CloudTrail.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExpectedNetworkServicePrincipal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/AWS Logs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22"
        },
        "BoolIfExists": {
          "aws:PrincipalIsAWSService": "false"
        }
      }
    }
  ]
}
```

Dans la vidéo suivante, découvrez comment utiliser la clé de condition `aws:PrincipalIsAWSService` dans une politique.

Accordez l'accès en toute sécurité à vos utilisateurs autorisés, aux emplacements réseau attendus et AWS aux services en même temps.

aws : PrincipalServiceName

Utilisez cette clé pour comparer le nom du principal de service dans la politique au principal de service qui effectue des demandes à vos ressources. Vous pouvez utiliser cette clé pour vérifier si cet appel est effectué par un principal de service spécifique. Lorsqu'un principal de

service effectue une demande directe à vos ressources, la clé `aws:PrincipalServiceName` contient le nom du principal de service. Par exemple, le nom principal du AWS CloudTrail service est`cloudtrail.amazonaws.com`.

- Disponibilité : cette clé est présente dans la demande lorsque l'appel est effectué par un responsable AWS de service. Cette clé n'est présente dans aucun autre cas, notamment les suivants :
  - Si le service utilise un rôle de service ou un rôle lié à un service pour effectuer un appel au nom du principal.
  - Si le service utilise les informations d'identification d'un principal IAM pour effectuer une demande au nom du principal.
  - Si l'appel est effectué directement par un principal IAM.
  - Si l'appel est passé par un demandeur anonyme.
- Types de données – Chaîne
- Type de valeur — Single-valued

Vous pouvez utiliser cette clé de condition pour limiter l'accès à vos identités fiables et aux emplacements réseau attendus tout en accordant l'accès à un AWS service en toute sécurité.

Dans l'exemple de politique de compartiment Amazon S3 suivant, l'accès au compartiment est restreint, sauf si la demande provient `vpc-111bbb22` ou provient d'un principal de service, tel que CloudTrail.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExpectedNetworkServicePrincipal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/AWS Logs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22",
          "aws:PrincipalServiceName": "cloudtrail.amazonaws.com"
```

```
        }
      }
    }
  ]
}
```

aws : PrincipalServiceNamesList

Cette clé fournit une liste de tous les noms des principal de service appartenant au service. Il s'agit d'une clé de condition avancée. Vous pouvez l'utiliser pour empêcher le service d'accéder à vos ressources à partir d'une région spécifique uniquement. Certains services peuvent créer des directeurs de service régionaux pour indiquer une instance particulière du service dans une région spécifique. Vous pouvez limiter l'accès à une ressource à une instance particulière du service. Lorsqu'un principal de service effectue une demande directe à vos ressources, `aws:PrincipalServiceNamesList` contient une liste non ordonnée de tous les noms des principaux de services associés à l'instance régionale du service.

- Disponibilité : cette clé est présente dans la demande lorsque l'appel est effectué par un responsable AWS de service. Cette clé n'est présente dans aucun autre cas, notamment les suivants :
  - Si le service utilise un rôle de service ou un rôle lié à un service pour effectuer un appel au nom du principal.
  - Si le service utilise les informations d'identification d'un principal IAM pour effectuer une demande au nom du principal.
  - Si l'appel est effectué directement par un principal IAM.
  - Si l'appel est passé par un demandeur anonyme.
- Type de données – Chaîne (liste)
- Type de valeur – À valeur multiple

`aws:PrincipalServiceNamesList` est une clé de condition à valeurs multiples. Les clés multivaluées peuvent avoir plusieurs valeurs dans le contexte de la demande. Vous devez utiliser les opérateurs d'ensemble `ForAnyValue` ou `ForAllValues` avec l'opérateur de condition de chaîne pour cette clé. Pour de plus amples informations sur les clés de condition à valeurs multiples, veuillez consulter Définir les opérateurs pour les clés de contexte à valeurs multiples.

aws : PrincipalType

Utilisez cette clé pour comparer le type de principal qui effectue la demande avec le type de principal spécifié dans la politique. Pour plus d'informations, veuillez consulter Comment spécifier un principal. Pour des exemples spécifiques de valeurs clés `principal`, veuillez consulter Valeurs de la clé du principal.

- Disponibilité : cette clé figure dans le contexte de la demande pour toutes les demandes, y compris les requêtes anonymes.
- Types de données – Chaîne
- Type de valeur — Single-valued

aws:userid

Utilisez cette clé pour comparer l'identifiant principal du demandeur avec l'ID spécifié dans la politique. Pour les utilisateurs IAM, la valeur du contexte de la demande est l'ID utilisateur. Pour les rôles IAM, ce format de valeur peut varier. Pour de plus amples informations sur l'ajout de principaux, veuillez consulter Comment spécifier un principal. Pour des exemples spécifiques de valeurs clés `principal`, veuillez consulter Valeurs de la clé du principal.

- Disponibilité : cette clé figure dans le contexte de la demande pour toutes les demandes, y compris les requêtes anonymes.
- Types de données – Chaîne
- Type de valeur — Single-valued

aws:username

Utilisez cette clé pour comparer le nom d'utilisateur du demandeur avec celui spécifié dans la politique. Pour de plus amples informations sur l'ajout de principaux, veuillez consulter Comment spécifier un principal. Pour des exemples spécifiques de valeurs clés `principal`, veuillez consulter Valeurs de la clé du principal.

- Availability (Disponibilité) : cette clé figure toujours dans le contexte de la demande pour les utilisateurs IAM. Les demandes anonymes et les requêtes effectuées à l'aide des rôles Utilisateur racine d'un compte AWS ou IAM n'incluent pas cette clé. Les requêtes effectuées à l'aide des informations d'identification IAM Identity Center n'incluent pas cette clé dans le contexte.
- Types de données – Chaîne

- Type de valeur — Single-valued

Propriétés d'une session de rôle

Utilisez les clés de condition suivantes pour comparer les propriétés de la session de rôle au moment de sa génération. Ces clés de condition ne sont disponibles que lorsqu'une demande est effectuée par un principal avec des informations d'identification de session de rôle ou de principal d'utilisateur fédéré. Les valeurs de ces clés de condition sont intégrées dans le jeton de session du rôle.

Un rôle est un type de principal. Vous pouvez également utiliser les clés de condition de la section Propriétés du principal pour évaluer les propriétés d'un rôle lorsque ce dernier effectue une requête.

aws : AssumedRoot

Utilisez cette touche pour vérifier si la demande a été faite en utilisant AssumeRoot. `AssumeRoot`renvoie des informations d'identification à court terme pour une session utilisateur root privilégiée que vous pouvez utiliser pour effectuer des actions privilégiées sur les comptes des membres de votre organisation. Pour de plus amples informations, veuillez consulter Gestion centralisée de l'accès root pour les comptes membres.

- Disponibilité : cette clé est incluse dans le contexte de la demande uniquement lorsque le principal utilise les informations d'identification AssumeRoot pour effectuer la demande.
- Types de données – Booléen
- Type de valeur — Single-valued

Dans l'exemple suivant, lorsqu'elle est utilisée comme politique de contrôle des services, elle refuse l'utilisation des informations d'identification à long terme d'un utilisateur root dans un compte AWS Organizations membre. La politique n'empêche pas les sessions `AssumeRoot` d'effectuer les actions autorisées par une session `AssumeRoot`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement":[
       {
          "Effect":"Deny",
          "Action":"*",
```

```
            "Resource": "*",
            "Condition":{
               "ArnLike":{
                  "aws:PrincipalArn":[
                     "arn:aws:iam::*:root"
                  ]
               },
               "Null":{
                  "aws:AssumedRoot":"true"
               }
            }
         }
      ]
   }
```

aws : FederatedProvider

Utilisez cette clé pour comparer le fournisseur d'identité (IdP) émetteur du principal avec l'IdP que vous spécifiez dans la politique. Cela signifie qu'un rôle IAM a été assumé à l'aide de l'`AssumeRoleWithWebIdentity` AWS STS opération. Lorsque les informations d'identification temporaires de la séance de rôle résultante sont utilisées pour effectuer une demande, le contexte de la demande identifie l'IdP qui a authentifié l'identité fédérée d'origine.

- Disponibilité : cette clé est présente dans la session de rôle d'un rôle qui a été assumé à l'aide du fournisseur OpenID Connect (OIDC) et dans la politique de confiance de rôle lorsqu'un fournisseur OIDC est utilisé pour appeler `AssumeRoleWithWebIdentity`.
- Types de données – Chaîne*
- Type de valeur — Single-valued

* Le type de données dépend de votre IdP :

- Si vous utilisez un AWS IdP intégré, tel qu'Amazon Cognito, la valeur clé sera une chaîne. La valeur clé peut ressembler à :`cognito-identity.amazonaws.com`.
- Si vous utilisez un IdP qui n'est pas intégré à AWS Amazon EKS, la valeur clé sera ARN. GitHub La valeur clé peut ressembler à :`arn:aws:iam::111122223333:oidc-provider/oidc.eks.region.amazonaws.com/id/OIDC_Provider_ID`.

Pour plus d'informations sur les IdPs réseaux externes`AssumeRoleWithWebIdentity`, consultezScénarios courants. Pour de plus amples informations, veuillez consulter Principaux de séance de rôle.

aws : TokenIssueTime

Utilisez cette clé pour comparer la date et l'heure d'émission des informations d'identification de sécurité temporaires avec celles spécifiées dans la politique.

- Availability (Disponibilité) : cette clé figure dans le contexte de la demande uniquement lorsque le principal utilise des informations d'identification temporaires pour effectuer la demande. La clé n'est pas présente dans AWS CLI les requêtes AWS d'API ou de AWS SDK effectuées à l'aide de clés d'accès.
- Type de données – Date
- Type de valeur — Single-valued

Pour savoir quels services prennent en charge l'utilisation d'informations d'identification temporaires, consultez AWS services qui fonctionnent avec IAM.

aws : SignInSessionArn

Utilisez cette clé pour comparer votre AWS Sign-In OAuth session avec celle que vous spécifiez dans la politique.

Lorsque vous utilisez un flux OAuth basé tel que AWS CLI login (`aws login`) ou AWS MCP Server, AWS Sign-In inclut un ARN de session de connexion dans les informations d'identification émises. AWS Sign-In propage cet ARN aux requêtes suivantes effectuées à l'aide de jetons OAuth d'accès. Cette clé vous permet de corréler l'activité de l'API avec la session de connexion d'origine et d'appliquer des politiques IAM à des sessions individuelles.

Pour les OAuth flux interactifs, le jeton d'actualisation détermine le cycle de vie de l'ARN de la session de connexion. Un nouveau jeton d'accès émis lors d'une actualisation continue de porter le même ARN de session de connexion.

Pour les OAuth flux non interactifs, le jeton d'accès détermine le cycle de vie de l'ARN de la session de connexion. Chaque nouveau jeton d'accès contient un nouvel ARN de session de connexion.

- Disponibilité : cette clé est incluse dans le contexte de la demande lorsque celle-ci provient d'une AWS Sign-In OAuth session. Cette clé n'est pas disponible pour les sessions de console.

- Type de données – ARN
- Type de valeur — Single-valued

Utilisez des opérateurs ARN plutôt que des opérateurs de chaîne lorsque vous comparez des ARN.

La politique suivante refuse toutes les actions associées à une AWS Sign-In session spécifique, même si les demandes provenant d'autres sessions actives se poursuivent :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenySpecificSignInSession",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
          "aws:SignInSessionArn": "arn:aws:signin:us-east-1:111122223333:session/
session-id"
        }
      }
    }
  ]
}
```

aws : MultiFactorAuthAge

Utilisez cette clé pour comparer le nombre de secondes écoulées depuis l'authentification du principal demandeur avec MFA avec le nombre spécifié dans la politique. Pour plus d'informations sur l'authentification MFA, consultez AWS Multi-factor authentification dans IAM.

> **Important**
>
> Cette clé de condition n'est pas présente pour les identités fédérées ou les demandes effectuées à l'aide de clés d'accès pour signer des requêtes AWS CLI, AWS API ou AWS SDK. Pour en savoir plus sur l'ajout d'une protection MFA aux opérations d'API à l'aide d'informations d'identification de sécurité temporaires, consultez Accès sécurisé aux API avec MFA.

Pour vérifier si le MFA est utilisé pour valider les identités fédérées IAM, vous pouvez transmettre la méthode d'authentification de votre fournisseur d'identité en AWS tant que balise de session. Pour en savoir plus, consultez Transmettre les balises de session AWS STS. Pour appliquer MFA aux identités IAM Identity Center, vous pouvez activer les attributs pour le contrôle d'accès afin de transmettre une demande d'assertion SAML avec la méthode d'authentification de votre fournisseur d'identité à IAM Identity Center.

- Disponibilité : cette clé figure dans le contexte de la demande uniquement lorsque le principal utilise des informations d'identification temporaires pour effectuer la requête. Les politiques avec des conditions MFA peuvent être attachées à ce qui suit :
  - Un utilisateur ou un groupe IAM
  - Une ressource telle qu'un compartiment Amazon S3, une file d'attente Amazon SQS ou une rubrique Amazon SNS
  - La politique d'approbation d'un rôle IAM qui peut être endossée par un utilisateur
- Type de données – Numérique
- Type de valeur — Single-valued

aws : MultiFactorAuthPresent

Utilisez cette clé pour vérifier si l'authentification multifactorielle (MFA) a été utilisée pour valider les informations d'identification de sécurité temporaires qui ont servi à effectuer la requête.

⚠ Important

Cette clé de condition n'est pas présente pour les identités fédérées ou les demandes effectuées à l'aide de clés d'accès pour signer des requêtes AWS CLI, AWS API ou AWS SDK. Pour en savoir plus sur l'ajout d'une protection MFA aux opérations d'API à l'aide d'informations d'identification de sécurité temporaires, consultez Accès sécurisé aux API avec MFA.

Pour vérifier si le MFA est utilisé pour valider les identités fédérées IAM, vous pouvez transmettre la méthode d'authentification de votre fournisseur d'identité en AWS tant que balise de session. Pour en savoir plus, consultez Transmettre les balises de session AWS STS. Pour appliquer MFA aux identités IAM Identity Center, vous pouvez activer les attributs

pour le contrôle d'accès afin de transmettre une demande d'assertion SAML avec la méthode d'authentification de votre fournisseur d'identité à IAM Identity Center.

- Availability (Disponibilité) : cette clé figure dans le contexte de la demande uniquement lorsque le principal utilise des informations d'identification temporaires pour effectuer la demande. Les politiques avec des conditions MFA peuvent être attachées à ce qui suit :
  - Un utilisateur ou un groupe IAM
  - Une ressource telle qu'un compartiment Amazon S3, une file d'attente Amazon SQS ou une rubrique Amazon SNS
  - La politique d'approbation d'un rôle IAM qui peut être endossée par un utilisateur
- Types de données – Booléen
- Type de valeur — Single-valued

Les informations d'identification temporaires sont utilisées pour authentifier les rôles IAM et les utilisateurs IAM à l'aide de jetons temporaires provenant de AssumeRole ou GetSessionToken, et les utilisateurs du. Console de gestion AWS

Les clés d'accès utilisateur IAM sont des informations d'identification à long terme, mais dans certains cas, elles AWS créent des informations d'identification temporaires pour le compte des utilisateurs IAM afin d'effectuer des opérations. Dans ces cas, la clé `aws:MultiFactorAuthPresent` est présente dans la demande et définie sur la valeur `false`. Deux scénarios courants peuvent expliquer ce comportement :

- Les utilisateurs IAM utilisent Console de gestion AWS sans le savoir des informations d'identification temporaires. Les utilisateurs se connectent à la console avec leur nom d'utilisateur et leur mot de passe, qui sont des informations d'identification à long terme. Toutefois, en arrière-plan, la console génère des informations d'identification temporaires pour le compte de l'utilisateur.
- Si un utilisateur IAM appelle un AWS service, celui-ci réutilise les informations d'identification de l'utilisateur pour adresser une autre demande à un autre service. Par exemple, lorsque vous appelez Athena pour accéder à un compartiment Amazon S3 ou lorsque vous l'utilisez CloudFormation pour créer une instance Amazon EC2. Pour la demande suivante, AWS utilise des informations d'identification temporaires.

Pour savoir quels services prennent en charge l'utilisation d'informations d'identification temporaires, consultez AWS services qui fonctionnent avec IAM.

La clé `aws:MultiFactorAuthPresent` n'est jamais présente lorsqu'une API ou une commande de l'interface de ligne de commande (CLI) est appelée avec des informations d'identification à long terme, telles que des paires de clés d'accès. Par conséquent, nous vous recommandons d'utiliser les versions `...IfExists` des opérateurs de condition lors de la vérification de cette clé.

Il est important de comprendre que l'élément `Condition` suivant ne constitue pas une méthode fiable pour vérifier si une demande est authentifiée avec MFA.

```
#####   WARNING: NOT RECOMMENDED   #####
"Effect" : "Deny",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Cette combinaison de l'effet `Deny`, de l'élément `Bool` et de la valeur `false` refuse les demandes qui peuvent être authentifiées à l'aide de MFA, mais ne l'ont pas été. Cela s'applique uniquement aux informations d'identification temporaires qui prennent en charge l'utilisation de l'authentification MFA. Cette instruction ne refuse pas l'accès aux demandes effectuées à l'aide d'informations d'identification à long terme, ni aux demandes qui ont été authentifiées avec MFA. Utilisez cet exemple avec prudence car sa logique est complexe et il ne permet pas de vérifier si elle MFA-authentication a été réellement utilisée.

De même, n'utilisez pas la combinaison de l'effet `Deny`, de l'élément `Null` et de `true`, car elle se comporte de la même manière et sa logique est encore plus complexe.

Combinaison recommandée

Nous vous recommandons plutôt d'utiliser l'opérateur `BoolIfExists` pour vérifier si une demande est authentifiée à l'aide de MFA.

```
"Effect" : "Deny",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Cette combinaison de `Deny`, `BoolIfExists` et `false` refuse les demandes qui ne sont pas authentifiées à l'aide de MFA. Elle refuse en particulier les demandes provenant d'informations d'identification temporaires qui n'incluent pas l'authentification MFA. Il refuse également les demandes effectuées à l'aide d'informations d'identification à long terme, telles que les opérations AWS CLI d' AWS API effectuées à l'aide de clés d'accès. L'opérateur `*IfExists` vérifie si la clé `aws:MultiFactorAuthPresent` existe et est présente ou non. Utilisez cet opérateur si vous

souhaitez refuser une demande qui n'est pas authentifiée à l'aide de MFA. C'est plus sûr, mais cela peut casser tout code ou script qui utilise des clés d'accès pour accéder à l' AWS API AWS CLI ou à l'API.

Combinaisons alternatives

Vous pouvez également utiliser l'`BoolIfExists`opérateur pour autoriser les requêtes AWS CLI et/ou les MFA-authenticated requêtes d' AWS API effectuées à l'aide d'informations d'identification à long terme.

```
"Effect" : "Allow",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "true" } }
```

Cette condition correspond à deux cas : si la clé existe et est présente ou si la clé n'existe pas. Cette combinaison de `Allow`, `BoolIfExists` et `true` autorise les demandes qui sont authentifiées à l'aide de MFA, ou les demandes qui ne peuvent pas être authentifiées à l'aide de MFA. Cela signifie que AWS CLI les opérations AWS d'API et de AWS SDK sont autorisées lorsque le demandeur utilise ses clés d'accès à long terme. Cette combinaison n'autorise pas les demandes provenant d'informations d'identification temporaires qui pourraient, mais n'incluent pas MFA.

Lorsque vous créez une politique à l'aide de l'éditeur visuel de la console IAM et que vous sélectionnez MFA required (MFA obligatoire), cette combinaison s'applique. Ce paramètre nécessite l'authentification MFA pour l'accès à la console, mais autorise un accès par programmation sans authentification MFA.

Sinon, vous pouvez utiliser l'opérateur `Bool` pour autoriser les demandes par programmation et les demandes de console uniquement lorsqu'elles sont authentifiées à l'aide de MFA.

```
"Effect" : "Allow",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "true" } }
```

Cette combinaison de `Allow``Bool`, et n'`true`autorise que les MFA-authenticated requêtes. Cela s'applique uniquement aux informations d'identification temporaires qui prennent en charge l'utilisation de l'authentification MFA. Cette instruction n'autorise pas l'accès aux demandes effectuées à l'aide de clés d'accès à long terme ou d'informations d'identification temporaires sans authentification MFA.

N'utilisez pas de construction de politique semblable à la suivante pour vérifier la présence de la clé MFA :

```
#####   WARNING: USE WITH CAUTION   #####

"Effect" : "Allow",
"Condition" : { "Null" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Cette combinaison de l'effet `Allow`, de l'élément `Null` et de la valeur `false` autorise uniquement les demandes qui peuvent être authentifiées à l'aide de MFA, que la demande soit réellement authentifiée ou non. Elle autorise toutes les demandes effectuées à l'aide d'informations d'identification temporaires et refuse l'accès aux informations d'identification à long terme. Utilisez cet exemple avec prudence car il ne permet pas de vérifier s'il MFA-authentication a été réellement utilisé.

aws : ChatbotSourceArn

Utilisez cette clé pour comparer l'ARN de configuration du chat source défini par le principal à l'ARN de configuration du chat que vous spécifiez dans la politique du rôle IAM associé à la configuration de votre canal. Vous pouvez autoriser les demandes en fonction de la session d'attribution de rôle initiée par Amazon Q Developer dans les applications de chat.

- Disponibilité : cette clé figure dans le contexte de la demande par le service Amazon Q Developer dans les applications de chat lorsqu'une session de rôle est prise en charge. La valeur clé est l'ARN de configuration du chat, par exemple lorsque vous exécutez une commande AWS CLI depuis un canal de discussion.
- Type de données – ARN
- Type de valeur — Single-valued
- Exemple de valeur : – `arn:aws::chatbot::123456789021:chat-configuration/slack-channel/private_channel`

La politique suivante refuse les requêtes put Amazon S3 put sur le compartiment spécifié pour toutes les requêtes provenant d'un canal Slack.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ExampleS3Deny",
```

```
            "Effect": "Deny",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
            "Condition": {
                "ArnLike": {
                      "aws:ChatbotSourceArn": "arn:aws:chatbot::*:chat-
configuration/slack-channel/*"
                }
            }
        }
    ]
}
```

aws : Ec2InstanceSourceVpc

Cette clé identifie le VPC auquel les informations d'identification du rôle IAM Amazon EC2 ont été fournies. Vous pouvez utiliser cette clé dans une politique avec la clé `aws:SourceVPC` globale pour vérifier si un appel provient d'un VPC (`aws:SourceVPC`) qui correspond au VPC auquel un identifiant a été délivré (`aws:Ec2InstanceSourceVpc`).

- Disponibilité : cette clé est incluse dans le contexte de la demande lorsque le demandeur signe des demandes avec des informations d'identification de rôle Amazon EC2. Elle peut être utilisée dans les politiques IAM, les politiques de contrôle des services, les politiques de points de terminaison d'un VPC et les politiques de ressources.
- Types de données – Chaîne
- Type de valeur — Single-valued

Cette clé peut être utilisée avec des valeurs d'identifiant VPC, mais elle est particulièrement utile lorsqu'elle est utilisée en tant que variable associée à la clé contextuelle `aws:SourceVpc`. Cette clé contextuelle `aws:SourceVpc` figure dans le contexte de la demande uniquement si le demandeur utilise un point de terminaison de VPC pour effectuer la demande. Utiliser `aws:Ec2InstanceSourceVpc` avec `aws:SourceVpc` permet d'utiliser `aws:Ec2InstanceSourceVpc` plus largement, car la comparaison porte sur des valeurs qui changent généralement ensemble.

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RequireSameVPC",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
            "aws:SourceVpc": "${aws:Ec2InstanceSourceVpc}"
        },
        "Null": {
          "ec2:SourceInstanceARN": "false"
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
}
```

Dans l'exemple ci-dessus, l'accès est refusé si la valeur `aws:SourceVpc` n'est pas égale à la valeur `aws:Ec2InstanceSourceVpc`. La déclaration de politique est limitée aux seuls rôles utilisés en tant que rôles d'instance Amazon EC2 en testant l'existence de la clé conditionnelle `ec2:SourceInstanceARN`.

La politique permet `aws:ViaAWSService` d' AWS autoriser les demandes lorsque les demandes sont effectuées au nom de vos rôles d'instance Amazon EC2. Par exemple, lorsque vous envoyez une demande depuis une instance Amazon EC2 à un compartiment Amazon S3 chiffré, Amazon S3 passe un appel AWS KMS en votre nom. Certaines clés ne sont pas présentes lorsque la demande est faite à AWS KMS.

aws : Ec2InstanceSourcePrivateIPv4

Cette clé identifie l'adresse IPv4 privée de l'elastic network interface principale à laquelle les informations d'identification du rôle IAM Amazon EC2 ont été fournies. Vous devez utiliser cette clé de condition avec sa clé associée `aws:Ec2InstanceSourceVpc` pour vous assurer que vous disposez d'une combinaison unique au monde d'ID VPC et d'adresse IP privée source. Utilisez cette clé pour vous `aws:Ec2InstanceSourceVpc` assurer qu'une demande a été faite à partir de la même adresse IP privée que celle à laquelle les informations d'identification ont été transmises.

- Disponibilité : cette clé est incluse dans le contexte de la demande lorsque le demandeur signe des demandes avec des informations d'identification de rôle Amazon EC2. Elle peut être utilisée dans les politiques IAM, les politiques de contrôle des services, les politiques de points de terminaison d'un VPC et les politiques de ressources.
- Type de données – Adresse IP
- Type de valeur — Single-valued

> **Important**
>
> Cette clé ne doit pas être utilisée seule dans une instruction `Allow`. Les adresses IP privées ne sont par définition pas uniques d'un point de vue global. Vous devez utiliser la clé `aws:Ec2InstanceSourceVpc` chaque fois que vous utilisez la clé `aws:Ec2InstanceSourcePrivateIPv4` pour spécifier le VPC à partir duquel les informations d'identification de votre instance Amazon EC2 peuvent être utilisées.

L'exemple suivant est une politique de contrôle des services (SCP) qui refuse l'accès à toutes les ressources, sauf si la demande arrive via un point de terminaison VPC dans le même VPC que les informations d'identification du rôle. Dans cet exemple, `aws:Ec2InstanceSourcePrivateIPv4` limite la source des informations d'identification à une instance particulière en fonction de l'adresse IP source.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action":  "*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:Ec2InstanceSourceVpc": "${aws:SourceVpc}"
                },
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
```

```
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action":  "*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:Ec2InstanceSourcePrivateIPv4": "${aws:VpcSourceIp}"
                },
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
                }
            }
        }
    ]
}
```

aws : SourceIdentity

Utilisez cette clé pour comparer l'identité source qui a été définie par le principal à l'identité source que vous spécifiez dans la politique.

- Disponibilité : cette clé est incluse dans le contexte de la demande une fois qu'une identité de source a été définie lorsqu'un rôle est assumé à l'aide d'une commande CLI AWS STS assume-role ou d'une opération d' AWS STS `AssumeRole`API.
- Types de données – Chaîne
- Type de valeur — Single-valued

Vous pouvez utiliser cette clé dans une politique pour autoriser les actions des donneurs AWS d'ordre qui ont défini une identité de source lorsqu'ils assument un rôle. L'activité pour l'identité source spécifiée du rôle apparaît dans AWS CloudTrail. Cela permet aux administrateurs de déterminer plus facilement qui a effectué des actions dans lesquelles un rôle a été attribué AWS.

Contrairement à `sts:RoleSessionName`, une fois l'identité source définie, la valeur ne peut plus être modifiée. Elle est présente dans le contexte de la demande pour toutes les actions entreprises par le rôle. La valeur persiste dans les sessions de rôle suivantes lorsque vous utilisez les informations d'identification de session pour endosser un autre rôle. Le fait d'endosser un rôle à partir d'un autre est appelé chaînage des rôles.

La `sts:SourceIdentity` clé est présente dans la demande lorsque le principal définit initialement une identité de source tout en assumant un rôle à l'aide d'une commande CLI AWS STS assume-role ou d'une opération d' AWS STS `AssumeRole`API. La clé `aws:SourceIdentity` est présente dans la demande pour toutes les actions qui sont effectuées avec une session de rôle disposant d'un ensemble d'identités source.

La politique de confiance de rôle suivante pour `CriticalRole` dans un compte `111122223333` contient une condition pour `aws:SourceIdentity` qui empêche un principal sans identité source définie sur Saanvi ou Diego d'endosser le rôle.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeRoleIfSourceIdentity",
            "Effect": "Allow",
            "Principal": {"AWS": "arn:aws:iam::123456789012:role/CriticalRole"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
                "StringLike": {
                    "aws:SourceIdentity": ["Saanvi","Diego"]
                }
            }
        }
    ]
}
```

Pour en savoir plus sur les informations relatives à l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.

EC2 : RoleDelivery

Utilisez cette clé pour comparer la version du service de métadonnées d'instance dans la requête signée avec les informations d'identification du rôle IAM pour Amazon EC2. Le service des métadonnées d'instance fait la distinction entre les demandes IMDSv1 et IMDSv2 pour une demande donnée en déterminant si les en-têtes `PUT` ou `GET`, qui sont propres à IMDSv2, sont présents dans cette demande.

- Disponibilité : cette clé est incluse dans le contexte de la requête chaque fois que la session de rôle est créée par une instance Amazon EC2.
- Type de données – Numérique
- Type de valeur — Single-valued
- Exemples de valeurs : 1,0, 2,0

Vous pouvez configurer le service des métadonnées d'instance (IMDS) sur chaque instance afin que le code local ou les utilisateurs locaux doivent utiliser IMDSv2. Lorsque vous spécifiez que IMDSv2 doit être utilisé, IMDSv1 ne fonctionne plus.

- Service de métadonnées d'instance, version 1 (IMDSv1) — Une méthode request/response
- Service des métadonnées d'instance Version 2 (IMDSv2) – méthode orientée session

Pour plus d'informations sur la configuration de votre instance pour l'utilisation d'IMDSv2, consultez la rubrique Configurer les options de métadonnées de l'instance.

Dans l'exemple suivant, l'accès est refusé si la RoleDelivery valeur ec2 : dans le contexte de la demande est 1.0 (IMDSv1). Cette instruction de politique peut être appliquée de manière générale car, si la demande n'est pas signée par les informations d'identification de rôle Amazon EC2, elle n'a aucun effet.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
            {
            "Sid": "RequireAllEc2RolesToUseV2",
            "Effect": "Deny",
            "Action": "*",
            "Resource": "*",
            "Condition": {
                "NumericLessThan": {
                    "ec2:RoleDelivery": "2.0"
                }
            }
        }
    ]
}
```

Pour plus d'informations, consultez Exemples de politiques pour travailler avec les métadonnées d'instance.

EC2 : SourceInstanceArn

Utilisez cette clé pour comparer l'ARN de l'instance à partir de laquelle la session du rôle est générée.

- Disponibilité : cette clé est incluse dans le contexte de la requête chaque fois que la session de rôle est créée par une instance Amazon EC2.
- Type de données – ARN
- Type de valeur — Single-valued
- Exemple de valeur : arn:aws:ec2:us-west - 2:111111111111 : -id instance/instance

Pour des exemples de politiques, consultez Accorder à une instance spécifique l'autorisation d'afficher des ressources dans d'autres services AWS.

colle : RoleAssumedBy

Le AWS Glue service définit cette clé de condition pour chaque demande d' AWS API où AWS Glue effectue une demande en utilisant un rôle de service pour le compte du client (pas par le biais d'un poste de travail ou d'un développeur, mais directement par le AWS Glue service). Utilisez cette clé pour vérifier si un appel à une AWS ressource provient du AWS Glue service.

- Disponibilité : cette clé est incluse dans le contexte de la demande lorsque AWS Glue vous faites une demande en utilisant un rôle de service pour le compte du client.
- Types de données – Chaîne
- Type de valeur — Single-valued
- Exemple de valeur : cette clé est toujours définie sur `glue.amazonaws.com`.

L'exemple suivant ajoute une condition pour permettre au AWS Glue service d'obtenir un objet depuis un compartiment Amazon S3.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:RoleAssumedBy": "glue.amazonaws.com"
        }
    }
}
```

colle : CredentialIssuingService

Le AWS Glue service définit cette clé pour chaque demande d' AWS API à l'aide d'un rôle de service provenant d'un poste de travail ou d'un point de terminaison de développeur. Utilisez cette clé pour vérifier si un appel à une AWS ressource provient d'un poste de AWS Glue travail ou d'un poste de développeur.

- Disponibilité : cette clé est incluse dans le contexte de la demande lorsqu'une demande AWS Glue provient d'un poste de travail ou d'un point de terminaison de développeur.
- Types de données – Chaîne
- Type de valeur — Single-valued
- Exemple de valeur : cette clé est toujours définie sur `glue.amazonaws.com`.

L'exemple suivant ajoute une condition attachée à un rôle IAM utilisé par une AWS Glue tâche. Cela garantit que certaines actions sont allowed/denied basées sur l'utilisation ou non de la session de rôle pour un environnement d'exécution des AWS Glue tâches.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:CredentialIssuingService": "glue.amazonaws.com"
        }
    }
}
```

construction de code : BuildArn

Cette clé identifie l'ARN de CodeBuild build auquel les informations d'identification des rôles IAM ont été transmises. Utilisez cette clé pour vérifier si un appel à une AWS ressource provient d'une CodeBuild version spécifique.

> **Note**
>
> La valeur complète de n'`codebuild:BuildArn`est pas connue à l'avance car elle contient l'ID de build généré dynamiquement.

- Disponibilité : cette clé est incluse dans le contexte de la demande chaque fois qu'une demande est faite par un rôle assumé par CodeBuild.
- Type de données – ARN
- Type de valeur — Single-valued
- Exemple de valeur : arn:aws:codebuild:us-east - 1:123456789012 ::12345678-1234-1234-1234-1234-123456789012 build/MyBuildProject

L'exemple suivant permet à une CodeBuild version spécifique d'`s3:GetObject`accéder au compartiment spécifié.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "ArnLike": {
```

```
            "codebuild:BuildArn": "arn:aws:codebuild:us-east-1:123456789012:build/
MyBuildProject:*"
        }
    }
}
```

construction de code : ProjectArn

Cette clé identifie l'ARN du CodeBuild projet auquel les informations d'identification des rôles IAM ont été fournies à une CodeBuild version. Utilisez cette clé pour vérifier si un appel à une AWS ressource provient d'un CodeBuild projet spécifique.

- Disponibilité : cette clé est incluse dans le contexte de la demande chaque fois qu'une demande est faite par un rôle assumé par CodeBuild.
- Type de données – ARN
- Type de valeur — Single-valued
- Exemple de valeur : arn:aws:codebuild:us-east - 1:123456789012 : project/MyBuildProject

L'exemple suivant permet à n'importe quel build d'un CodeBuild projet spécifique d'`s3:GetObject`accéder au compartiment spécifié.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "ArnEquals": {
            "codebuild:ProjectArn": "arn:aws:codebuild:us-east-1:123456789012:project/
MyBuildProject"
        }
    }
}
```

lambda : SourceFunctionArn

Utilisez cette clé pour identifier l'ARN de la fonction Lambda auquel les informations d'identification du rôle IAM ont été fournies. Le service Lambda définit cette clé pour chaque demande AWS d'API provenant de l'environnement d'exécution de votre fonction. Utilisez cette clé pour vérifier si un appel à une AWS ressource provient du code d'une fonction Lambda spécifique. Lambda définit également

cette clé pour certaines requêtes provenant de l'extérieur de l'environnement d'exécution, telles que l'écriture de journaux CloudWatch et l'envoi de traces à X-Ray.

- Disponibilité : cette clé est incluse dans le contexte de la requête chaque fois que le code de la fonction Lambda est invoqué.
- Type de données – ARN
- Type de valeur — Single-valued
- Exemple de valeur : arn:aws:lambda:us-east - 1:123456789012:function : TestFunction

L'exemple suivant permet à une fonction Lambda spécifique d'accéder `s3:PutObject` au compartiment spécifié.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ExampleSourceFunctionArn",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
            "Condition": {
                "ArnEquals": {
                    "lambda:SourceFunctionArn": "arn:aws:lambda:us-
east-1:123456789012:function:source_lambda"
                }
            }
        }
    ]
}
```

Pour plus d'informations, consultez la section Travailler avec les informations d'identification de l'environnement d'exécution Lambda dans le Guide du développeur AWS Lambda .

sms : SourceInstanceArn

Utilisez cette clé pour identifier l'ARN de l'instance AWS Systems Manager gérée à laquelle les informations d'identification du rôle IAM ont été transmises. Cette clé de condition n'est pas présente

lorsque la requête provient d'une instance gérée avec un rôle IAM associé à un profil d'instance Amazon EC2.

- Disponibilité : cette clé est incluse dans le contexte de la requête lorsque les informations d'identification des rôles sont fournies à une instance AWS Systems Manager gérée.
- Type de données – ARN
- Type de valeur — Single-valued
- Exemple de valeur : arn:aws:ec2:us-west - 2:111111111111 : -id instance/instance

boutique d'identité : UserId

Utilisez cette clé pour comparer l'identité du personnel d'IAM Identity Center dans la requête signée avec l'identité spécifiée dans la politique.

- Disponibilité : cette clé est incluse lorsque l'appelant de la requête est un utilisateur d'IAM Identity Center.
- Types de données – Chaîne
- Type de valeur — Single-valued
- Exemple de valeur : 94482488-3041-7026-18f3-be45837cd0e4

Vous pouvez trouver le nom UserId d'un utilisateur dans IAM Identity Center en adressant une demande à l'GetUserIdAPI à l'aide de l' AWS CLI AWS API ou du AWS SDK.

Propriétés du réseau

Utilisez les clés de condition suivantes pour comparer les détails du réseau d'où provient la requête ou par lesquels elle est passée avec les propriétés du réseau que vous spécifiez dans la politique.

aws : SourceIp

Utilisez cette clé pour comparer l'adresse IP du demandeur avec celle spécifiée dans la politique. La clé de condition `aws:SourceIp` ne peut être utilisée que pour les plages d'adresses IP publiques.

- Availability (Disponibilité) : cette clé figure dans le contexte de la demande, sauf lorsque le demandeur utilise un point de terminaison VPC pour effectuer la demande.
- Type de données – Adresse IP
- Type de valeur — Single-valued

La clé de condition `aws:SourceIp` peut être utilisée dans une politique pour autoriser des principaux à effectuer des demandes uniquement dans une plage IP spécifiée.

> **Note**
>
> `aws:SourceIp` prend en charge des adresses ou une plage d'adresses IP IPv4 et IPv6. Pour obtenir la liste de ceux Services AWS qui prennent en charge IPv6, consultez Services AWS la section relative à la prise en charge du protocole IPv6 dans le guide de l'utilisateur Amazon VPC.

Par exemple, vous pouvez attacher la politique basée sur l'identité suivante à un rôle IAM. Cette politique permet à l'utilisateur de placer des objets dans le compartiment Amazon S3 `amzn-s3-demo-bucket3` si l'appel est effectué à partir de la plage d'adresses IPv4 spécifiée. Cette politique autorise également un AWS service qui Transmission des sessions d'accès effectue cette opération en votre nom.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket3/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
        }
    ]
}
```

Si vous devez restreindre l'accès depuis les réseaux qui prennent en charge à la fois l'adressage IPv4 et IPv6, vous pouvez inclure l'adresse ou les plages d'adresses IPv4 et IPv6 dans la condition de politique IAM. La politique basée sur l'identité suivante permet à l'utilisateur de placer

des objets dans le compartiment Amazon S3 `amzn-s3-demo-bucket3` si l'utilisateur effectue l'appel à partir des plages d'adresses IPv4 ou IPv6 spécifiées. Avant d'inclure des plages d'adresses IPv6 dans votre politique IAM, vérifiez que l'adresse avec laquelle Service AWS vous travaillez est compatible avec IPv6. Pour obtenir la liste de ceux Services AWS qui prennent en charge IPv6, consultez Services AWS la section relative à la prise en charge du protocole IPv6 dans le guide de l'utilisateur Amazon VPC.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket3/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": [
                        "203.0.113.0/24",
                        "2001:DB8:1234:5678::/64"
                    ]
                }
            }
        }
    ]
}
```

Si la demande provient d'un hôte qui utilise un point de terminaison Amazon VPC, la clé `aws:SourceIp` n'est pas disponible. Vous devriez plutôt utiliser une VPC-specific clé telle que aws : VpcSourceIp. Pour plus d'informations sur l'utilisation des points de terminaison d'un VPC, veuillez consulter la rubrique Gestion des identités et des accès pour les points de terminaison d'un VPC et les services de points de terminaison d'un VPC dans le Guide AWS PrivateLink .

> **Note**
>
> Lorsque vous Services AWS passez des appels à d'autres Services AWS personnes en votre nom (appels de service à service), certains contextes d'autorisation spécifiques au réseau

sont supprimés. Si votre politique utilise cette clé de condition avec des `Deny` instructions, il se peut que Service AWS les principaux soient bloqués involontairement. Pour permettre aux Services AWS de fonctionner correctement tout en respectant vos exigences en matière de sécurité, excluez les principaux services de vos instructions `Deny` en ajoutant la clé de condition `aws:PrincipalIsAWSService` avec la valeur `false`.

aws : SourceVpc

Utilisez cette clé pour vérifier si une requête passe par le VPC auquel le point de terminaison de VPC est attaché. Dans une politique, vous pouvez utiliser cette clé pour autoriser l'accès à un seul VPC spécifique. Pour de plus amples informations, veuillez consulter Restriction de l'accès à un VPC spécifique dans le Guide de l'utilisateur service de stockage simple Amazon.

- Availability (Disponibilité) : cette clé figure dans le contexte de la demande uniquement si le demandeur utilise un point de terminaison de VPC pour effectuer la demande.
- Types de données – Chaîne
- Type de valeur — Single-valued

Dans une politique, vous pouvez utiliser cette clé pour autoriser ou restreindre l'accès à un VPC spécifique.

Par exemple, vous pouvez associer la politique basée sur l'identité suivante à un rôle IAM afin de refuser au compartiment `amzn-s3-demo-bucket3` Amazon S3, sauf si la demande est faite `PutObject` à partir de l'ID VPC spécifié ou par Services AWS ce dernier utilise des sessions d'accès direct (FAS) pour effectuer des demandes au nom du rôle. Contrairement à aws : SourceIp, vous devez utiliser aws:ViaAWSService ou aws : CalledVia pour autoriser les requêtes FAS, car le VPC source de la requête initiale n'est pas préservé.

> **Note**
>
> Cette politique ne permet aucune action. Utilisez cette stratégie conjointement à d'autres stratégies qui autorisent des actions spécifiques.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "PutObjectIfNotVPCID",
      "Effect": "Deny",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket3/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-1234567890abcdef0"
        },
        "Bool": {
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
}
```

> **Note**
>
> AWS recommande de l'utiliser `aws:SourceVpcArn` plutôt que `aws:SourceVpc` s'il `aws:SourceVpcArn` est pris en charge par le service que vous ciblez. Consultez aws : SourceVpcArn pour obtenir la liste des services pris en charge.

> **Note**
>
> Lorsque vous Services AWS passez des appels à d'autres Services AWS personnes en votre nom (appels de service à service), certains contextes d'autorisation spécifiques au réseau sont supprimés. Si votre politique utilise cette clé de condition avec des `Deny` instructions, il se peut que Service AWS les principaux soient bloqués involontairement. Pour permettre aux Services AWS de fonctionner correctement tout en respectant vos exigences en matière de sécurité, excluez les principaux services de vos instructions `Deny` en ajoutant la clé de condition `aws:PrincipalIsAWSService` avec la valeur `false`.

aws : SourceVpcArn

Utilisez cette clé pour vérifier l'ARN du VPC par lequel une demande a été transmise via un point de terminaison VPC. Cette clé renvoie l'ARN du VPC auquel le point de terminaison du VPC est attaché.

- Disponibilité : cette clé est incluse dans le contexte de demande pour les services pris en charge lorsqu'une demande est effectuée via un point de terminaison VPC. La clé n'est pas incluse pour les demandes effectuées via des points de terminaison de service public. Les services suivants prennent en charge cette clé :
  - AWS App Runner (préfixe : `apprunner`)
  - AWS Application Discovery Service (préfixe : `discovery`)
  - Amazon Athena (prefixe : `athena`)
  - AWS Cloud Map (préfixe : `servicediscovery`)
  - Amazon CloudWatch Application Insights (préfixe : `applicationinsights`)
  - AWS CloudFormation (préfixe : `cloudformation`)
  - Amazon Comprehend Medical (prefix: `comprehendmedical`)
  - AWS Optimiseur de calcul (préfixe :) `compute-optimizer`
  - Amazon ; Elastic Container Registry (préfixe : `ecr`)
  - Amazon Elastic Container Service (préfixe : `ecs`)
  - Amazon Kinesis Analytics (préfixe :) `kinesisanalytics`
  - Amazon Route 53 (préfixe : `route53`)
  - AWS DataSync (préfixe : `datasync`)
  - Amazon Elastic Block Store (prefixe : `ebs`)
  - Amazon EventBridge Scheduler (préfixe :) `scheduler`
  - Amazon Data Firehose (prefixe : `firehose`)
  - AWS HealthImaging (préfixe : `medical-imaging`)
  - AWS HealthLake (préfixe : `healthlake`)
  - AWS HealthOmics (préfixe : `omics`)
  - Gestion des identités et des accès AWS (sauf pour l'`iam:PassRole`action) (préfixe : `iam`)
  - AWS IoT FleetWise (préfixe : `iotfleetwise`)
  - AWS IoT Wireless (préfixe : `iotwireless`)
  - AWS Key Management Service (préfixe : `kms`)

- AWS Lambda (préfixe : `lambda`)
- AWS Payment Cryptography (préfixe : `payment-cryptography`)
- Amazon Polly (prefixe : `polly`)
- Autorité de certification privée AWS (préfixe : `acm-pca`)
- AWS Corbeille (préfixe : `rbin`)
- Amazon Rekognition (prefixe : `rekognition`)
- Service Quotas (préfixe : `servicequotas`)
- Amazon Simple Storage Service (prefixe : `s3`)
- AWS Storage Gateway (préfixe : `storagegateway`)
- AWS Systems Manager Incident Manager Contacts (préfixe : `ssm-contacts`)
- Amazon Textract (prefixe : `textract`)
- Amazon Transcribe (prefixe : `transcribe`)
- AWS Transfer Family (préfixe : `transfer`)

- Type de données : ARN

  AWS recommande d'utiliser des opérateurs ARN plutôt que des opérateurs de chaîne lors de la comparaison des ARN.

- Type de valeur — Single-valued
- Exemple de valeur : – `arn:aws:ec2:us-east-1:123456789012:vpc/vpc-0e9801d129EXAMPLE`

Voici un exemple de politique de bucket qui refuse l'accès à `amzn-s3-demo-bucket` et à ses objets depuis n'importe quel endroit en dehors du VPC`vpc-1a2b3c4d`.

```
{
   "Version":"2012-10-17",
   "Statement": [
     {
       "Sid": "Access-to-specific-VPC-only",
       "Principal": "*",
       "Action": "s3:*",
       "Effect": "Deny",
       "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket",
                    "arn:aws:s3:::amzn-s3-demo-bucket/*"],
       "Condition": {
```

```
          "ArnNotEquals": {
            "aws:SourceVpcArn": "arn:aws:ec2:us-east-1:*:vpc/vpc-1a2b3c4d"
          }
        }
      }
    ]
}
```

> **Note**
>
> Lorsque vous Services AWS passez des appels à d'autres Services AWS personnes en votre nom (appels de service à service), certains contextes d'autorisation spécifiques au réseau sont supprimés. Si votre politique utilise cette clé de condition avec des `Deny` instructions, il se peut que Service AWS les principaux soient bloqués involontairement. Pour permettre aux Services AWS de fonctionner correctement tout en respectant vos exigences en matière de sécurité, excluez les principaux services de vos instructions `Deny` en ajoutant la clé de condition `aws:PrincipalIsAWSService` avec la valeur `false`.

aws : SourceVpce

Utilisez cette clé pour comparer l'identifiant du point de terminaison de VPC de la demande avec l'ID du point de terminaison spécifié dans la politique.

- Availability (Disponibilité) : cette clé figure dans le contexte de la demande uniquement si le demandeur utilise un point de terminaison de VPC pour effectuer la demande.
- Types de données – Chaîne
- Type de valeur — Single-valued

Dans une politique, vous pouvez utiliser cette clé pour restreindre l'accès à un point de terminaison de VPC spécifique. Pour de plus amples informations, consultez Restriction de l'accès à un VPC spécifique dans le Guide de l'utilisateur service de stockage simple Amazon. De la même manière que vous utilisezaws : SourceVpc, vous devez utiliser aws:ViaAWSService ou aws : CalledVia autoriser les demandes effectuées à l' Services AWS aide de sessions d'accès direct (FAS). En effet, le point de terminaison de VPC source de la requête initiale n'est pas conservé.

> **Note**
>
> Lorsque vous Services AWS passez des appels à d'autres Services AWS personnes en votre nom (appels de service à service), certains contextes d'autorisation spécifiques au réseau sont supprimés. Si votre politique utilise cette clé de condition avec des `Deny` instructions, il se peut que Service AWS les principaux soient bloqués involontairement. Pour permettre aux Services AWS de fonctionner correctement tout en respectant vos exigences en matière de sécurité, excluez les principaux services de vos instructions `Deny` en ajoutant la clé de condition `aws:PrincipalIsAWSService` avec la valeur `false`.

aws : VpceAccount

Utilisez cette clé pour comparer l'ID de AWS compte auquel appartient le point de terminaison VPC via lequel la demande a été effectuée avec l'ID de compte que vous spécifiez dans la politique. Cette clé de condition vous aide à établir des contrôles du périmètre réseau en garantissant que les demandes transitent par des points de terminaison d'un VPC appartenant à des comptes spécifiques.

- Disponibilité : cette clé est incluse dans le contexte de la demande lorsqu'une demande est effectuée via un point de terminaison d'un VPC. La clé n'est pas incluse pour les demandes effectuées via des points de terminaison de service public.

  Les services suivants prennent en charge cette clé :

  - AWS App Runner (préfixe : `apprunner`)
  - AWS Application Discovery Service (préfixe : `discovery`)
  - Amazon Athena (prefixe : `athena`)
  - AWS Cloud Map (préfixe : `servicediscovery`)
  - Amazon CloudWatch Application Insights (préfixe : `applicationinsights`)
  - AWS CloudFormation (préfixe : `cloudformation`)
  - Amazon Comprehend Medical (prefix: `comprehendmedical`)
  - AWS Optimiseur de calcul (préfixe :) `compute-optimizer`
  - Amazon ; Elastic Container Registry (préfixe : `ecr`)
  - Amazon Elastic Container Service (préfixe : `ecs`)
  - Amazon Kinesis Analytics (préfixe :) `kinesisanalytics`
  - Amazon Route 53 (préfixe : `route53`)

- AWS DataSync (préfixe : `datasync`)
- Amazon DynamoDB (préfixe :) `dynamodb`
- Amazon Elastic Block Store (prefixe : `ebs`)
- Amazon EventBridge Scheduler (préfixe :) `scheduler`
- Amazon Data Firehose (prefixe : `firehose`)
- Amazon GameLift (préfixe : `gamelift`)
- AWS HealthImaging (préfixe : `medical-imaging`)
- AWS HealthLake (préfixe : `healthlake`)
- AWS HealthOmics (préfixe : `omics`)
- Gestion des identités et des accès AWS (sauf pour l'`iam:PassRole`action) (préfixe : `iam`)
- AWS IoT FleetWise (préfixe : `iotfleetwise`)
- AWS IoT Wireless (préfixe : `iotwireless`)
- AWS Key Management Service (préfixe : `kms`)
- AWS Lambda (préfixe : `lambda`)
- AWS Payment Cryptography (préfixe : `payment-cryptography`)
- Amazon Polly (prefixe : `polly`)
- Autorité de certification privée AWS (préfixe : `acm-pca`)
- AWS Corbeille (préfixe : `rbin`)
- Amazon Rekognition (prefixe : `rekognition`)
- Service Quotas (préfixe : `servicequotas`)
- Amazon Simple Storage Service (prefixe : `s3`)
- AWS Bouclier (préfixe : `shield`)
- AWS Storage Gateway (préfixe : `storagegateway`)
- AWS Systems Manager Incident Manager Contacts (préfixe : `ssm-contacts`)
- Amazon Textract (prefixe : `textract`)
- Amazon Transcribe (prefixe : `transcribe`)
- AWS Transfer Family (préfixe : `transfer`)

- Types de données – Chaîne
- Type de valeur – Single-valued
- Exemple de valeur : – `123456789012`

Vous pouvez utiliser cette clé de condition pour restreindre l'accès aux ressources afin que les demandes passent par les points de terminaison d'un VPC appartenant à votre compte. L'exemple de stratégie de compartiment Amazon S3 suivant autorise l'accès lorsque la demande provient d'un point de terminaison d'un VPC appartenant au compte spécifié :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AccessToSpecificVpceAccountOnly",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/RoleName"
            },
            "Action": "s3:GetObject",
            "Effect": "Allow",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/*",
            "Condition": {
                "StringEquals": {
                    "aws:VpceAccount": "111122223333"
                }
            }
        }
    ]
}
```

> **Note**
>
> Cette clé de condition est actuellement prise en charge pour un ensemble de AWS services sélectionnés. L'utilisation de cette clé avec des services non pris en charge peut entraîner des résultats d'autorisation imprévus. Délimitez toujours la clé de condition aux services pris en charge dans vos politiques.

Certains AWS services accèdent à vos ressources depuis leurs réseaux lorsqu'ils agissent en votre nom. Si vous utilisez de tels services, vous devrez modifier l'exemple de politique ci-dessus pour permettre aux AWS services d'accéder à vos ressources depuis l'extérieur de votre réseau. Pour plus d'informations sur les modèles d'accès à prendre en compte lors de l'application de contrôles

d'accès basés sur l'origine de la demande, consultez Mettez en place des barrières de protection des autorisations à l'aide de périmètres de données.

aws : VpceOrg ID

Utilisez cette clé pour comparer l'identifiant de l'organisation propriétaire du point de terminaison VPC à partir duquel la demande a été effectuée avec l'identifiant que vous spécifiez dans la politique. AWS Organizations Cette clé de condition offre l'approche la plus évolutive en matière de contrôles du périmètre réseau, en incluant automatiquement tous les points de terminaison d'un VPC appartenant aux comptes de votre organisation.

- Disponibilité : cette clé est incluse dans le contexte de la demande lorsqu'une demande est effectuée via un point de terminaison VPC et que le compte propriétaire du terminal VPC est membre d'une AWS organisation. La clé n'est pas incluse pour les demandes effectuées via d'autres chemins réseau ou lorsque le compte propriétaire du point de terminaison d'un VPC ne fait pas partie d'une organisation.

  Les services suivants prennent en charge cette clé :

  - AWS App Runner (préfixe : `apprunner`)
  - AWS Application Discovery Service (préfixe : `discovery`)
  - Amazon Athena (prefixe : `athena`)
  - AWS B2B Échange de données (préfixe : `b2bi`)
  - AWS Cloud Map (préfixe : `servicediscovery`)
  - AWS CloudHSM (préfixe : `cloudhsmv2`)
  - Amazon CloudWatch Application Insights (préfixe : `applicationinsights`)
  - AWS CloudFormation (préfixe : `cloudformation`)
  - Amazon Cognito (préfixe :) `cognito-identity`
  - Amazon Comprehend Medical (prefix: `comprehendmedical`)
  - AWS Optimiseur de calcul (préfixe :) `compute-optimizer`
  - AWS Database Migration Service (préfixe : `dms`)
  - AWS Données du service d'annuaire (préfixe : `ds-data`)
  - Amazon DynamoDB (préfixe :) `dynamodb`
  - Amazon ; Elastic Container Registry (préfixe : `ecr`)
  - Amazon Elastic Container Service (préfixe : `ecs`)
  - Flux de données Amazon Kinesis (préfixe :) `kinesis`

- Amazon Kinesis Analytics (préfixe :) `kinesisanalytics`
- Amazon Q Business (préfixe : `qbusiness`)
- Amazon Route 53 (préfixe : `route53`)
- AWS DataSync (préfixe : `datasync`)
- Amazon Elastic Block Store (prefixe : `ebs`)
- Amazon EventBridge Scheduler (préfixe :) `scheduler`
- Amazon Data Firehose (prefixe : `firehose`)
- Amazon GameLift (préfixe : `gamelift`)
- AWS HealthImaging (préfixe : `medical-imaging`)
- AWS HealthLake (préfixe : `healthlake`)
- AWS HealthOmics (préfixe : `omics`)
- Gestion des identités et des accès AWS (sauf pour l'`iam:PassRole`action) (préfixe : `iam`)
- AWS Identity Store (préfixe : `identitystore`)
- AWS IoT FleetWise (préfixe : `iotfleetwise`)
- AWS IoT TwinMaker (préfixe : `iottwinmaker`)
- AWS IoT Wireless (préfixe : `iotwireless`)
- Amazon Keyspaces (pour Apache Cassandra) (préfixe :) `cassandra`
- AWS Key Management Service (préfixe : `kms`)
- AWS Lambda (préfixe : `lambda`)
- AWS Network Firewall (préfixe : `network-firewall`)
- AWS Payment Cryptography (préfixe : `payment-cryptography`)
- Service SMS et voix Amazon Pinpoint (préfixe :) `sms-voice`
- Amazon Polly (prefixe : `polly`)
- AWS Price List (préfixe : `pricing`)
- Autorité de certification privée AWS (préfixe : `acm-pca`)
- AWS Corbeille (préfixe : `rbin`)
- Amazon Rekognition (prefixe : `rekognition`)
- Service Quotas (préfixe : `servicequotas`)
- Amazon Simple Email Service (préfixe : `ses`)
- AWS Bouclier (préfixe : `shield`)

- Amazon Simple Storage Service (prefixe : `s3`)
- Amazon Simple Queue Service (préfixe : `sqs`)
- AWS Storage Gateway (préfixe : `storagegateway`)
- AWS Systems Manager Incident Manager Contacts (préfixe : `ssm-contacts`)
- Amazon Textract (prefixe : `textract`)
- Amazon Transcribe (prefixe : `transcribe`)
- AWS Transfer Family (préfixe : `transfer`)
- Amazon WorkMail (préfixe : `workmail`)

- Types de données – Chaîne
- Type de valeur — Single-valued
- Exemples de valeurs — `o-a1b2c3d4e5`

L'exemple de politique de contrôle des ressources suivant refuse l'accès à votre Amazon S3 et à vos AWS Key Management Service ressources, sauf si la demande provient de points de terminaison VPC appartenant à l'organisation spécifiée ou de réseaux de AWS services agissant en votre nom. Certaines organisations peuvent avoir besoin de modifier davantage cette politique pour répondre à leurs besoins, par exemple, autoriser l'accès à des partenaires tiers. Pour plus d'informations sur les modèles d'accès à prendre en compte lors de l'application de contrôles d'accès basés sur l'origine de la demande, consultez Mettez en place des barrières de protection des autorisations à l'aide de périmètres de données.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "EnforceNetworkPerimeterVpceOrgID",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:*",
        "kms:*"
      ],
      "Resource": "*",
      "Condition": {
```

```
        "BoolIfExists": {
          "aws:PrincipalIsAWSService": "false",
          "aws:ViaAWSService": "false"
        },
        "StringNotEqualsIfExists": {
            "aws:VpceOrgID": "o-abcdef0123",
            "aws:PrincipalTag/network-perimeter-exception": "true"
        }
      }
    }
  ]
}
```

> **Note**
>
> Cette clé de condition est actuellement prise en charge pour un ensemble de AWS services sélectionnés. L'utilisation de cette clé avec des services non pris en charge peut entraîner des résultats d'autorisation imprévus. Délimitez toujours la clé de condition aux services pris en charge dans vos politiques.

> **Note**
>
> Lorsque vous Services AWS passez des appels à d'autres Services AWS personnes en votre nom (appels de service à service), certains contextes d'autorisation spécifiques au réseau sont supprimés. Si votre politique utilise cette clé de condition avec des `Deny` instructions, il se peut que Service AWS les principaux soient bloqués involontairement. Pour permettre aux Services AWS de fonctionner correctement tout en respectant vos exigences en matière de sécurité, excluez les principaux services de vos instructions `Deny` en ajoutant la clé de condition `aws:PrincipalIsAWSService` avec la valeur `false`.

aws : VpceOrgPaths

Utilisez cette clé pour comparer le AWS Organizations chemin du point de terminaison VPC à partir duquel la demande a été faite avec le chemin que vous spécifiez dans la politique. Cette clé de condition vous permet de mettre en œuvre des contrôles du périmètre réseau au niveau de l'unité

organisationnelle (UO), en s'adaptant automatiquement à l'utilisation de votre point de terminaison d'un VPC lorsque vous ajoutez de nouveaux points de terminaison dans les UO spécifiées.

- Disponibilité : cette clé est incluse dans le contexte de la demande lorsqu'une demande est effectuée via un point de terminaison d'un VPC et que le compte propriétaire du point de terminaison d'un VPC est membre d'une organisation. La clé n'est pas incluse pour les demandes effectuées via d'autres chemins réseau ou lorsque le compte propriétaire du point de terminaison d'un VPC ne fait pas partie d'une organisation.

  Les services suivants prennent en charge cette clé :

  - AWS App Runner (préfixe : `apprunner`)
  - AWS Application Discovery Service (préfixe : `discovery`)
  - Amazon Athena (prefixe : `athena`)
  - AWS Cloud Map (préfixe : `servicediscovery`)
  - Amazon CloudWatch Application Insights (préfixe : `applicationinsights`)
  - AWS CloudFormation (préfixe : `cloudformation`)
  - Amazon Comprehend Medical (prefix: `comprehendmedical`)
  - AWS Optimiseur de calcul (préfixe :) `compute-optimizer`
  - Amazon ; Elastic Container Registry (préfixe : `ecr`)
  - Amazon Elastic Container Service (préfixe : `ecs`)
  - Amazon Kinesis Analytics (préfixe :) `kinesisanalytics`
  - Amazon Route 53 (préfixe : `route53`)
  - AWS DataSync (préfixe : `datasync`)
  - Amazon DynamoDB (préfixe :) `dynamodb`
  - Amazon Elastic Block Store (prefixe : `ebs`)
  - Amazon EventBridge Scheduler (préfixe :) `scheduler`
  - Amazon Data Firehose (prefixe : `firehose`)
  - Amazon GameLift (préfixe : `gamelift`)
  - AWS HealthImaging (préfixe : `medical-imaging`)
  - AWS HealthLake (préfixe : `healthlake`)
  - AWS HealthOmics (préfixe : `omics`)
  - Gestion des identités et des accès AWS (sauf pour l'`iam:PassRole`action) (préfixe : `iam`)

- AWS IoT FleetWise (préfixe : `iotfleetwise`)
- AWS IoT Wireless (préfixe : `iotwireless`)
- AWS Key Management Service (préfixe : `kms`)
- AWS Lambda (préfixe : `lambda`)
- AWS Payment Cryptography (préfixe : `payment-cryptography`)
- Amazon Polly (prefixe : `polly`)
- Autorité de certification privée AWS (préfixe : `acm-pca`)
- AWS Corbeille (préfixe : `rbin`)
- Amazon Rekognition (prefixe : `rekognition`)
- Service Quotas (préfixe : `servicequotas`)
- AWS Secrets Manager (préfixe : `secretsmanager`)
- AWS Security Token Service (préfixe : `sts`)
- Amazon Simple Storage Service (prefixe : `s3`)
- AWS Bouclier (préfixe : `shield`)
- AWS Storage Gateway (préfixe : `storagegateway`)
- AWS Systems Manager Incident Manager Contacts (préfixe : `ssm-contacts`)
- Amazon Textract (prefixe : `textract`)
- Amazon Transcribe (prefixe : `transcribe`)
- AWS Transfer Family (préfixe : `transfer`)

- Type de données – Chaîne (liste)
- Type de valeur – À valeur multiple
- Exemples de valeurs — `o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/`

Comme `aws:VpceOrgPaths`est une clé de condition à valeurs multiples, vous devez utiliser les opérateurs `ForAnyValue` et `ForAllValues` avec des opérateurs de condition de chaîne pour cette clé. L'exemple suivant de stratégie de compartiment Amazon S3 autorise l'accès uniquement lorsque les demandes proviennent de points de terminaison d'un VPC appartenant à des comptes dans des unités organisationnelles spécifiques :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessFromSpecificOrgPaths",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/RoleName"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/*",
      "Condition": {
        "ForAnyValue:StringLike": {
          "aws:VpceOrgPaths": [
            "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/*",
            "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-33333333/*"
          ]
        }
      }
    }
  ]
}
```

> **Note**
>
> Cette clé de condition est actuellement prise en charge pour un ensemble de AWS services sélectionnés. L'utilisation de cette clé avec des services non pris en charge peut entraîner des résultats d'autorisation imprévus. Délimitez toujours la clé de condition aux services pris en charge dans vos politiques.

Certains AWS services accèdent à vos ressources depuis leurs réseaux lorsqu'ils agissent en votre nom. Si vous utilisez de tels services, vous devrez modifier l'exemple de politique ci-dessus pour permettre aux AWS services d'accéder à vos ressources depuis l'extérieur de votre réseau. Pour plus d'informations sur les modèles d'accès à prendre en compte lors de l'application de contrôles d'accès basés sur l'origine de la demande, consultez Mettez en place des barrières de protection des autorisations à l'aide de périmètres de données.

aws : VpcSourceIp

Utilisez cette clé pour comparer l'adresse IP à partir de laquelle une demande a été effectuée avec celle spécifiée dans la politique. Dans une politique, la clé ne correspond que si la demande provient de l'adresse IP spécifiée et qu'elle passe par un point de terminaison de VPC.

- Availability (Disponibilité) : cette clé ne figure dans le contexte de la demande que si la demande est effectuée à l'aide d'un point de terminaison de VPC.
- Type de données – Adresse IP
- Type de valeur — Single-valued

Pour plus d'informations, consultez Contrôle de l'accès aux points de terminaison d'un VPC à l'aide de politiques de point de terminaison dans le Guide de l'utilisateur Amazon VPC. De la même manière que vous utilisezaws : SourceVpc, vous devez utiliser aws:ViaAWSService ou aws : CalledVia autoriser les demandes effectuées à l' Services AWS aide de sessions d'accès direct (FAS). En effet, l'adresse IP source de la requête initiale effectuée à l'aide d'un point de terminaison de VPC n'est pas conservée dans les requêtes FAS.

> **Note**
>
> `aws:VpcSourceIp` prend en charge des adresses ou une plage d'adresses IP IPv4 et IPv6. Pour obtenir la liste de ceux Services AWS qui prennent en charge IPv6, consultez Services AWS la section relative à la prise en charge du protocole IPv6 dans le guide de l'utilisateur Amazon VPC.
> Utilisez toujours la touche de `aws:VpcSourceIp` condition avec les touches de `aws:SourceVpcArn` condition `aws:SourceVpcaws:SourceVpce`, ou. Dans le cas contraire, une politique peut autoriser les appels d'API depuis un VPC inattendu qui utilise le même CIDR IP ou qui se chevauche. Cela peut se produire parce que les CIDR IP de deux VPC indépendants peuvent être identiques ou se chevaucher.

> **Note**
>
> Lorsque vous Services AWS passez des appels à d'autres Services AWS personnes en votre nom (appels de service à service), certains contextes d'autorisation spécifiques au réseau sont supprimés. Si votre politique utilise cette clé de condition avec des `Deny` instructions, il se peut que Service AWS les principaux soient bloqués involontairement. Pour permettre

aux Services AWS de fonctionner correctement tout en respectant vos exigences en matière de sécurité, excluez les principaux services de vos instructions `Deny` en ajoutant la clé de condition `aws:PrincipalIsAWSService` avec la valeur `false`.

Propriétés de la ressource

Utilisez les clés de condition suivantes pour comparer les détails de la ressource cible de la requête avec les propriétés de la ressource que vous spécifiez dans la politique.

aws : ResourceAccount

Utilisez cette clé pour comparer l'ID de l'Compte AWS du propriétaire de la ressource demandée avec le compte de la ressource dans la politique. Vous pouvez ensuite autoriser ou refuser l'accès à cette ressource en fonction du compte propriétaire de la ressource.

- Disponibilité : cette clé figure toujours dans le contexte de la demande pour la plupart des services. Les actions suivantes ne prennent pas en charge cette clé :
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - AWS Directory Service
    - `ds:AcceptSharedDirectory`
  - Amazon Elastic Block Store : toutes les actions
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CreateTransitGatewayPeeringAttachment`
    - `ec2:CreateVpcEndpoint`
    - `ec2:CreateVpcPeeringConnection`
  - Amazon EventBridge
    - `events:PutEvents`— EventBridge `PutEvents` appelle un bus événementiel sur un autre compte, si ce bus événementiel a été configuré comme EventBridge cible intercomptes avant

le 2 mars 2023. Pour plus d'informations, consultez la section Accorder des autorisations pour autoriser des événements provenant d'autres AWS comptes dans le Guide de EventBridge l'utilisateur Amazon.

- Amazon GuardDuty
  - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
  - `macie2:AcceptInvitation`
- OpenSearch Service Amazon
  - `es:AcceptInboundConnection`
- Amazon Route 53
  - `route53:AssociateVpcWithHostedZone`
  - `route53:CreateVPCAssociationAuthorization`
- AWS Security Hub CSPM
  - `securityhub:AcceptAdministratorInvitation`

- Types de données – Chaîne
- Type de valeur — Single-valued

> **Note**
>
> Pour des considérations supplémentaires concernant les actions non prises en charge ci-dessus, consultez le référentiel Exemples de politiques relatives aux périmètres de données (français non garanti).

Cette clé est égale à l' Compte AWS ID du compte contenant les ressources évaluées dans la demande.

Pour la plupart des ressources de votre compte, l'ARN contient l'ID du compte propriétaire de cette ressource. Pour certaines ressources, telles que les compartiments Amazon S3, l'ARN de ressource n'inclut pas l'ID de compte. Les deux exemples suivants montrent la différence entre une ressource avec un ID de compte dans l'ARN et un ARN Amazon S3 sans ID de compte :

- `arn:aws:iam::123456789012:role/AWSExampleRole` – rôle IAM créé et détenu dans le compte 123456789012.

- `arn:aws:s3:::amzn-s3-demo-bucket2` : compartiment Amazon S3 créé et détenu dans le compte `111122223333`, non affiché dans l'ARN.

Utilisez la AWS console, l'API ou la CLI pour trouver toutes vos ressources et les ARN correspondants.

Vous élaborez une politique qui refuse les autorisations d'accès aux ressources en fonction de l'ID de compte du propriétaire de la ressource. Par exemple, la politique basée sur l'identité suivante refuse l'accès à la ressource spécifiée si celle-ci n'appartient pas au compte spécifié.

Pour utiliser cette politique, remplacez le texte en italique de l'espace réservé dans l'exemple de politique par vos propres informations de compte.

> **Important**
>
> Cette politique ne permet aucune action. Au lieu de cela, elle utilise l'effet `Deny` qui refuse explicitement l'accès à toutes les ressources répertoriées dans l'instruction n'appartenant pas au compte répertorié. Utilisez cette politique en combinaison avec d'autres politiques qui autorisent l'accès à des ressources spécifiques.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyInteractionWithResourcesNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:us-east-1:111122223333:*"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
```

```
      }
    }
  ]
}
```

Cette politique refuse l'accès à toutes les ressources pour un AWS service spécifique, sauf si la personne spécifiée Compte AWS est propriétaire de la ressource.

> **Note**
>
> Certains Services AWS nécessitent l'accès à des ressources AWS détenues qui sont hébergées dans un autre Compte AWS. L'utilisation de `aws:ResourceAccount` dans vos politiques basées sur l'identité peut avoir un impact sur la capacité de votre identité à accéder à ces ressources.

Certains AWS services, tels que AWS Data Exchange, s'appuient sur l'accès à des ressources extérieures à la vôtre Comptes AWS pour les opérations normales. Si vous utilisez l'élément `aws:ResourceAccount` dans vos politiques, incluez des déclarations supplémentaires afin de créer des dérogations pour ces services. L'exemple de politique AWS: Refusez l'accès aux ressources Amazon S3 en dehors de votre compte, sauf AWS Data Exchange montre comment refuser l'accès en fonction du compte de la ressource tout en définissant des exceptions pour les ressources appartenant au service.

Utilisez cet exemple de politique comme modèle pour créer vos propres politiques personnalisées. Reportez-vous à votre documentation de service pour plus d'informations.

aws : ResourceOrgPaths

Utilisez cette clé pour comparer le AWS Organizations chemin de la ressource consultée au chemin indiqué dans la politique. Dans une politique, cette clé de condition vérifie que le demandeur est un membre du compte au sein de l'organisation racine ou des unités organisationnelles (UO) spécifiées dans AWS Organizations. Un AWS Organizations chemin est une représentation textuelle de la structure d'une entité Organisations. Pour de plus amples informations sur l'utilisation et la compréhension des chemins, veuillez consulter Comprenez les AWS Organizations chemin de l'entité.

- Disponibilité : cette clé ne figure dans le contexte de la demande que si le compte qui possède la ressource est membre d'une organisation. Cette clé de condition globale ne prend pas en charge les actions suivantes :
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - AWS Directory Service
    - `ds:AcceptSharedDirectory`
  - Amazon Elastic Block Store : toutes les actions
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CreateTransitGatewayPeeringAttachment`
    - `ec2:CreateVpcEndpoint`
    - `ec2:CreateVpcPeeringConnection`
  - Amazon EventBridge
    - `events:PutEvents`— EventBridge `PutEvents` appelle un bus événementiel sur un autre compte, si ce bus événementiel a été configuré comme EventBridge cible intercomptes avant le 2 mars 2023. Pour plus d'informations, consultez la section Accorder des autorisations pour autoriser des événements provenant d'autres AWS comptes dans le Guide de EventBridge l'utilisateur Amazon.
  - Amazon GuardDuty
    - `guardduty:AcceptAdministratorInvitation`
  - Amazon Macie
    - `macie2:AcceptInvitation`
  - OpenSearch Service Amazon
    - `es:AcceptInboundConnection`
  - Amazon Route 53
    - `route53:AssociateVpcWithHostedZone`

    - `route53:CreateVPCAssociationAuthorization`
  - AWS Security Hub CSPM
    - `securityhub:AcceptAdministratorInvitation`
- Type de données – Chaîne (liste)
- Type de valeur – À valeur multiple

> **Note**
>
> Pour des considérations supplémentaires concernant les actions non prises en charge ci-dessus, consultez le référentiel Exemples de politiques relatives aux périmètres de données (français non garanti).

`aws:ResourceOrgPaths` est une clé de condition à valeurs multiples. Les clés multivaluées peuvent avoir plusieurs valeurs dans le contexte de la demande. Vous devez utiliser les opérateurs d'ensemble `ForAnyValue` ou `ForAllValues` avec l'opérateur de condition de chaîne pour cette clé. Pour de plus amples informations sur les clés de condition à valeurs multiples, veuillez consulter Définir les opérateurs pour les clés de contexte à valeurs multiples.

Par exemple, la condition suivante renvoie `True` pour les ressources qui appartiennent à l'organisation `o-a1b2c3d4e5`. Lorsque vous incluez un caractère générique, vous devez utiliser l'opérateur de condition StringLike.

```
"Condition": {
      "ForAnyValue:StringLike": {
            "aws:ResourceOrgPaths":["o-a1b2c3d4e5/*"]
   }
}
```

La condition suivante renvoie `True` pour les ressources avec l'ID d'unité d'organisation `ou-ab12-11111111`. Elle correspond aux ressources détenues par les comptes rattachés à l'unité d'organisation ou-ab12-11111111 ou à l'une des unités d'organisation enfants.

```
"Condition": { "ForAnyValue:StringLike" : {
     "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/*"]
}}
```

La condition suivante renvoie `True` pour les ressources détenues par des comptes rattachés directement à l'ID d'unité d'organisation `ou-ab12-22222222`, mais pas aux unités d'organisation enfants. L'exemple suivant utilise l'opérateur de StringEquals condition pour spécifier l'exigence de correspondance exacte pour l'ID de l'unité d'organisation et non une correspondance par caractères génériques.

```
"Condition": { "ForAnyValue:StringEquals" : {
     "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
}}
```

> **Note**
>
> Certains Services AWS nécessitent l'accès à des ressources AWS détenues qui sont hébergées dans un autre Compte AWS. L'utilisation de `aws:ResourceOrgPaths` dans vos politiques basées sur l'identité peut avoir un impact sur la capacité de votre identité à accéder à ces ressources.

Certains AWS services, tels que AWS Data Exchange, s'appuient sur l'accès à des ressources extérieures à la vôtre Comptes AWS pour les opérations normales. Si vous utilisez la clé `aws:ResourceOrgPaths` dans vos politiques, incluez des déclarations supplémentaires afin de créer des dérogations pour ces services. L'exemple de politique AWS: Refusez l'accès aux ressources Amazon S3 en dehors de votre compte, sauf AWS Data Exchange montre comment refuser l'accès en fonction du compte de la ressource tout en définissant des exceptions pour les ressources appartenant au service. Vous pouvez créer une politique similaire pour restreindre l'accès aux ressources d'une unité d'organisation (UO) à l'aide de la clé `aws:ResourceOrgPaths`, tout en tenant compte des ressources appartenant au service.

Utilisez cet exemple de politique comme modèle pour créer vos propres politiques personnalisées. Reportez-vous à votre documentation de service pour plus d'informations.

aws : ResourceOrg ID

Utilisez cette clé pour comparer l'identifiant de l'organisation AWS Organizations à laquelle appartient la ressource demandée avec l'identifiant spécifié dans la politique.

- Disponibilité : cette clé ne figure dans le contexte de la demande que si le compte qui possède la ressource est membre d'une organisation. Cette clé de condition globale ne prend pas en charge les actions suivantes :

- AWS Audit Manager
  - `auditmanager:UpdateAssessmentFrameworkShare`
- Amazon Detective
  - `detective:AcceptInvitation`
- AWS Directory Service
  - `ds:AcceptSharedDirectory`
- Amazon Elastic Block Store : toutes les actions
- Amazon EC2
  - `ec2:AcceptTransitGatewayPeeringAttachment`
  - `ec2:AcceptVpcEndpointConnections`
  - `ec2:AcceptVpcPeeringConnection`
  - `ec2:CreateTransitGatewayPeeringAttachment`
  - `ec2:CreateVpcEndpoint`
  - `ec2:CreateVpcPeeringConnection`
- Amazon EventBridge
  - `events:PutEvents`— EventBridge `PutEvents` appelle un bus événementiel sur un autre compte, si ce bus événementiel a été configuré comme EventBridge cible intercomptes avant le 2 mars 2023. Pour plus d'informations, consultez la section Accorder des autorisations pour autoriser des événements provenant d'autres AWS comptes dans le Guide de EventBridge l'utilisateur Amazon.
- Amazon GuardDuty
  - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
  - `macie2:AcceptInvitation`
- OpenSearch Service Amazon
  - `es:AcceptInboundConnection`
- Amazon Route 53
  - `route53:AssociateVpcWithHostedZone`
  - `route53:CreateVPCAssociationAuthorization`
- AWS Security Hub CSPM
  - `securityhub:AcceptAdministratorInvitation`

- Types de données – Chaîne
- Type de valeur — Single-valued

> **Note**
>
> Pour des considérations supplémentaires concernant les actions non prises en charge ci-dessus, consultez le référentiel Exemples de politiques relatives aux périmètres de données (français non garanti).

Cette clé globale renvoie l'ID de l'organisation de la ressource pour une demande donnée. Elle vous permet de créer des règles qui s'appliquent à toutes les ressources d'une organisation qui sont spécifiées dans l'élément `Resource` d'une politique basée sur l'identité. Vous pouvez spécifier l'ID de l'organisation dans l'élément Condition. Lorsque vous ajoutez et supprimez des comptes, les politiques qui contiennent la clé `aws:ResourceOrgID` incluent automatiquement les bons comptes et vous n'avez pas besoin de la mettre à jour manuellement.

Par exemple, la politique suivante empêche le principal d'ajouter des objets à la ressource `policy-genius-dev`, sauf si la ressource Amazon S3 appartient à la même organisation que le principal qui effectue la demande.

> **Important**
>
> Cette politique ne permet aucune action. Au lieu de cela, elle utilise l'effet `Deny` qui refuse explicitement l'accès à toutes les ressources répertoriées dans l'instruction n'appartenant pas au compte répertorié. Utilisez cette politique en combinaison avec d'autres politiques qui autorisent l'accès à des ressources spécifiques.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Sid": "DenyPutObjectToS3ResourcesOutsideMyOrganization",
        "Effect": "Deny",
        "Action": "s3:PutObject",
        "Resource": "arn:aws:s3:::policy-genius-dev/*",
```

```
        "Condition": {
            "StringNotEquals": {
                "aws:ResourceOrgID": "${aws:PrincipalOrgID}"
            }
        }
    }
}
```

> **Note**
>
> Certains Services AWS nécessitent l'accès à des ressources AWS détenues qui sont hébergées dans un autre Compte AWS. L'utilisation de `aws:ResourceOrgID` dans vos politiques basées sur l'identité peut avoir un impact sur la capacité de votre identité à accéder à ces ressources.

Certains AWS services, tels que AWS Data Exchange, s'appuient sur l'accès à des ressources extérieures à la vôtre Comptes AWS pour les opérations normales. Si vous utilisez la clé `aws:ResourceOrgID` dans vos politiques, incluez des déclarations supplémentaires afin de créer des dérogations pour ces services. L'exemple de politique AWS: Refusez l'accès aux ressources Amazon S3 en dehors de votre compte, sauf AWS Data Exchange montre comment refuser l'accès en fonction du compte de la ressource tout en définissant des exceptions pour les ressources appartenant au service. Vous pouvez créer une politique similaire pour restreindre l'accès aux ressources de votre organisation à l'aide de la clé `aws:ResourceOrgID`, tout en tenant compte des ressources appartenant au service.

Utilisez cet exemple de politique comme modèle pour créer vos propres politiques personnalisées. Reportez-vous à votre documentation de service pour plus d'informations.

Dans la vidéo suivante, découvrez comment utiliser la clé de condition `aws:ResourceOrgID` dans une politique.

Assurez-vous que les identités et les réseaux ne peuvent être utilisés que pour accéder à des ressources fiables.

les lois :ResourceTag/clé-balise

Utilisez cette clé pour comparer la paire valeur clé d'étiquette que vous spécifiez dans la politique avec la paire valeur clé attachée à la ressource. Par exemple, vous pouvez exiger que l'accès à une

ressource soit autorisé uniquement si la clé de balise `"Dept"` est attachée à la ressource avec la valeur `"Marketing"`. Pour de plus amples informations, veuillez consulter Contrôle de l'accès aux AWS ressources.

- Disponibilité : cette clé figure dans le contexte de la demande lorsque la ressource demandée possède déjà des balises attachées ou dans les demandes qui créent une ressource avec une balise attachée. Cette clé est renvoyée uniquement pour les ressources qui prennent en charge l'autorisation basée sur les balises. Il y a une clé de contexte pour chaque paire clé-valeur de balise.
- Types de données – Chaîne
- Type de valeur — Single-valued

Cette clé de contexte est formatée `"aws:ResourceTag/`*`tag-key`*`":"`*`tag-value`*`"` où *`tag-key`* et *`tag-value`* constituent une paire clé de balise et valeur. Les clés de balises ne sont pas sensibles à la casse. Cela signifie que si vous spécifiez `"aws:ResourceTag/TagKey1": "Value1"` dans l'élément de condition de votre politique, la condition correspond à une clé de balise de ressource nommée `TagKey1` ou `tagkey1`, mais pas aux deux. Les valeurs de ces key/value paires de balises distinguent les majuscules et minuscules. Cela signifie que si vous spécifiez `"aws:ResourceTag/TagKey1": "Production"` dans l'élément de condition de votre politique, la condition correspond à une clé de balise de ressource nommée `Production`, mais elle ne correspond pas à `production` ou `PRODUCTION`.

Pour obtenir des exemples d'utilisation de la clé `aws:ResourceTag` pour contrôler l'accès aux ressources IAM, consultez la section Contrôle de l'accès aux AWS ressources.

Pour des exemples d'utilisation de la `aws:ResourceTag` touche pour contrôler l'accès à d'autres AWS ressources, consultezContrôle de l'accès aux AWS ressources à l'aide de balises.

Pour obtenir un didacticiel sur l'utilisation de la clé de condition `aws:ResourceTag` pour le contrôle d'accès basé sur les attributs (ABAC), reportez-vous à la section Tutoriel IAM : définir les autorisations d'accès AWS ressources basées sur des balises.

Propriétés de la requête

Utilisez les clés de condition suivantes pour comparer les détails de la requête elle-même et le contenu de la requête avec les propriétés de la requête que vous spécifiez dans la politique.

aws : CalledVia

Utilisez cette clé pour comparer les services de la politique avec les services qui ont créé des demandes au nom du principal IAM (utilisateur ou rôle). Lorsqu'un principal adresse une demande à un AWS service, ce service peut utiliser les informations d'identification du principal pour effectuer des demandes ultérieures à d'autres services. Lorsque la demande est effectuée à l'aide de sessions d'accès direct (FAS), cette clé est définie avec la valeur du principal de service. La clé `aws:CalledVia` contient une liste ordonnée des services de la chaîne ayant effectué des demandes pour le compte du principal.

Pour de plus amples informations, veuillez consulter Transmission des sessions d'accès.

- Availability (Disponibilité): cette clé est présente dans la demande lorsqu'un service prenant en charge `aws:CalledVia` utilise les informations d'identification d'un principal IAM pour effectuer une demande à un autre service. Cette clé n'est pas présente si le service utilise un rôle de service ou un rôle lié à un service pour effectuer un appel au nom du principal. Cette clé n'est pas non plus présente lorsque le principal effectue directement l'appel.
- Type de données – Chaîne (liste)
- Type de valeur – À valeur multiple

Pour utiliser la clé de `aws:CalledVia` condition dans une politique, vous devez fournir les principaux de service pour autoriser ou refuser les demandes AWS de service. Par exemple, vous pouvez l'utiliser AWS CloudFormation pour lire et écrire à partir d'une table Amazon DynamoDB. DynamoDB utilise ensuite le chiffrement fourni par AWS Key Management Service ()AWS KMS.

Pour autoriser ou refuser l'accès lorsqu'un service effectue une demande à l'aide des informations d'identification du principal, utilisez la clé de condition `aws:ViaAWSService`. Cette clé de condition soutient les AWS services.

La clé `aws:CalledVia` est une clé à valeurs multiples. Cependant, vous ne pouvez pas imposer l'ordre en utilisant cette clé dans une condition. Dans l'exemple ci-dessus, User 1 (Utilisateur 1) effectue une demande à CloudFormation, qui appelle DynamoDB, qui appelle AWS KMS. Il s'agit de trois demandes distinctes. Le dernier appel à AWS KMS est effectué par l'utilisateur 1 via DynamoDB, CloudFormation puis par DynamoDB.
Exemple utilisant AWS :CalledVia.

Dans ce cas, la clé `aws:CalledVia` dans le contexte de la demande comprend `cloudformation.amazonaws.com` et `dynamodb.amazonaws.com`, dans cet ordre. Si vous vous

souciez uniquement du fait que l'appel a été passé via DynamoDB quelque part dans la chaîne de demandes, vous pouvez utiliser cette clé de condition dans votre politique.

Par exemple, la politique suivante permet de gérer la AWS KMS clé nomméemy-example-key, mais uniquement si DynamoDB est l'un des services demandeurs. L'opérateur de condition ForAnyValue:StringEquals s'assure que DynamoDB est un des services appelant. Si le principal effectue directement l'appel à AWS KMS , la condition renvoie false et la demande n'est pas autorisée par cette politique.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaDynamodb",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/my-example-key",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:CalledVia": [
                        "dynamodb.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

Si vous souhaitez définir le service effectuant le premier ou le dernier appel de la chaîne, vous pouvez utiliser les clés aws:CalledViaLast et aws:CalledViaFirst. Par exemple, la politique suivante permet de gérer la clé nommée my-example-key dans AWS KMS. Ces AWS KMS opérations ne sont autorisées que si plusieurs demandes ont été incluses dans la chaîne. La

première demande doit être faite via DynamoDB CloudFormation et la dernière via DynamoDB. Si d'autres services font des demandes au milieu de la chaîne, l'opération est toujours autorisée.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaChain",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/my-example-key",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaFirst": "cloudformation.amazonaws.com",
                    "aws:CalledViaLast": "dynamodb.amazonaws.com"
                }
            }
        }
    ]
}
```

Les clés `aws:CalledViaFirst` et `aws:CalledViaLast` sont présentes dans la demande lorsqu'un service utilise les informations d'identification d'un principal IAM pour appeler un autre service. Elles indiquent le premier et le dernier services ayant effectué des appels dans la chaîne de demandes. Supposons, par exemple, que cela CloudFormation appelle un autre service nomméX `Service`, qui appelle DynamoDB, qui appelle ensuite. AWS KMS Le dernier appel à AWS KMS est effectué par `User 1` through CloudFormation`X Service`, puis DynamoDB. Il a été appelé pour la première fois CloudFormation et appelé pour la dernière fois via DynamoDB.
Exemple utilisant aws : CalledViaFirst et aws :CalledViaLast.

aws : CalledViaFirst

Utilisez cette clé pour comparer les services de la politique avec le premier service ayant fait une demande au nom du principal IAM (utilisateur ou rôle). Pour de plus amples informations, veuillez consulter `aws:CalledVia`.

- Availability (Disponibilité) : cette clé est présente dans la requête lorsqu'un service utilise les informations d'identification d'un principal IAM pour effectuer au moins une autre requête à un service différent. Cette clé n'est pas présente si le service utilise un rôle de service ou un rôle lié à un service pour effectuer un appel au nom du principal. Cette clé n'est pas non plus présente lorsque le principal effectue directement l'appel.
- Types de données – Chaîne
- Type de valeur — Single-valued

aws : CalledViaLast

Utilisez cette clé pour comparer les services de la politique avec le dernier service ayant fait une demande au nom du principal IAM (utilisateur ou rôle). Pour plus d'informations, veuillez consulter `aws:CalledVia`.

- Availability (Disponibilité) : cette clé est présente dans la requête lorsqu'un service utilise les informations d'identification d'un principal IAM pour effectuer au moins une autre requête à un service différent. Cette clé n'est pas présente si le service utilise un rôle de service ou un rôle lié à un service pour effectuer un appel au nom du principal. Cette clé n'est pas non plus présente lorsque le principal effectue directement l'appel.
- Types de données – Chaîne
- Type de valeur — Single-valued

aws:ViaAWSService

Utilisez cette touche pour vérifier si un utilisateur adresse Service AWS une demande à un autre service en votre nom à l'aide de sessions d'accès direct (FAS).

La clé de contexte de la requête renvoie `true` lorsqu'un service utilise des sessions d'accès direct pour effectuer une requête au nom du principal IAM d'origine. La clé de contexte de demande renvoie également `false` lorsque le principal effectue l'appel directement.

- Availability (Disponibilité) : cette clé figure toujours dans le contexte de la demande.

- Types de données – Booléen
- Type de valeur — Single-valued

aws : ViaCustomerDomain

Utilisez cette touche pour vérifier si un responsable a fait une demande en utilisant les informations d'identification obtenues auprès de la délégation d'accès temporaire (TAD). Lorsqu'un partenaire ou un fournisseur obtient des informations d'identification temporaires par le biais d'un échange de jetons TAD, nous définissons cette clé sur l'identifiant de compte du partenaire demandeur.

Utilisez cette clé de condition dans les politiques de gestion des AWS identités et des accès (IAM) pour étendre les contrôles d'accès spécifiquement aux sessions établies par une délégation d'accès temporaire, sans affecter vos propres sessions. Par exemple, vous pouvez utiliser cette clé pour révoquer uniquement les sessions basées sur la délégation tout en préservant l'accès direct à la console.

- Disponibilité : cette clé est présente dans le contexte de la demande lorsqu'un principal utilise des informations d'identification obtenues via un échange de jetons TAD.
- Types de données – Chaîne
- Type de valeur — Single-valued

L'exemple suivant montre une politique de refus pour les sessions obtenues par délégation d'accès temporaire à partir d'un compte partenaire spécifique. La politique s'applique uniquement aux sessions publiées avant une heure spécifiée. Cet exemple utilise un caractère générique (*) pour `Action` et `Resource` dans l'`Deny`instruction, qui est le modèle standard pour révoquer tous les accès d'une session spécifique. Pour révoquer uniquement des actions spécifiques, remplacez le caractère générique par la liste d'actions appropriée.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "*",
            "Resource": "*",
            "Condition": {
                "DateLessThan": {
                    "aws:TokenIssueTime": "2026-07-30T01:00:37.819Z"
```

```
                },
                "StringEquals": {
                    "aws:ViaCustomerDomain": "123456789012"
                }
            }
        }
    ]
}
```

en tant que : CalledVia AWSMCP

Utilisez cette clé pour comparer les services de la politique avec les services AWS MCP qui ont effectué des demandes pour le compte du principal IAM (utilisateur ou rôle). Lorsqu'un principal adresse une demande à un service AWS MCP, ce service utilise les informations d'identification du principal pour effectuer des demandes ultérieures à d'autres services. Lorsque la demande est faite à l'aide d'un service AWS MCP, cette clé est définie avec la valeur du principal de service. La `aws:CalledViaAWSMCP` clé contient le nom principal du service MCP qui a effectué des demandes au nom du principal.

- Disponibilité : cette clé est présente dans la demande lorsqu'un service AWS MCP utilise les informations d'identification d'un principal IAM pour adresser une demande à un AWS service. Cette clé n'est pas non plus présente lorsque le principal effectue directement l'appel.
- Types de données – Chaîne
- Type de valeur — Single-valued

Vous pouvez utiliser cette clé de condition pour autoriser ou refuser l'accès en fonction du serveur MCP spécifique à l'origine de la demande. Par exemple, la politique suivante interdit les opérations de suppression sensibles lorsqu'elles sont initiées via un serveur MCP spécifique :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenySensitiveActionsViaSpecificMCP",
            "Effect": "Deny",
            "Action": [
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "dynamodb:DeleteTable"
            ],
```

```
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaAWSMCP": "aws-mcp.amazonaws.com"
                }
            }
        }
    ]
}
```

AWS : via le service AWS SMCP

Utilisez cette touche pour vérifier si un service AWS MCP envoie une demande à un autre AWS service en votre nom à l'aide de sessions d'accès direct (FAS). La clé de contexte de demande `true` est renvoyée lorsqu'un service AWS MCP transmet une demande à un AWS service pour le compte du principal IAM d'origine. La clé de contexte de demande renvoie également `false` lorsque le principal effectue l'appel directement.

- Disponibilité : cette clé est incluse dans le contexte de demande lorsqu'un serveur AWS MCP adresse une demande à un AWS service en aval pour le compte d'un principal IAM.
- Types de données – Booléen
- Type de valeur — Single-valued

Vous pouvez utiliser cette touche pour restreindre des actions spécifiques lorsqu'elles passent par les serveurs MCP. Par exemple, la politique suivante interdit les opérations de suppression sensibles lorsqu'elles sont initiées via un serveur AWS MCP :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenySensitiveActionsViaMCP",
            "Effect": "Deny",
            "Action": [
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "dynamodb:DeleteTable"
            ],
            "Resource": "*",
            "Condition": {
```

```
                "Bool": {
                    "aws:ViaAWSMCPService": "true"
                }
            }
        }
    ]
}
```

aws : CurrentTime

Utilisez cette clé pour comparer la date et l'heure de la demande avec celles spécifiée dans la politique. Pour visualiser un exemple de politique qui utilise cette clé de condition, veuillez consulter AWS : permet l'accès en fonction de la date et de l'heure.

- Availability (Disponibilité) : cette clé figure toujours dans le contexte de la demande.
- Type de données – Date
- Type de valeur — Single-valued

aws : EpochTime

Utilisez cette clé pour comparer la date et l'heure de la demande au format Epoch ou Unix avec la valeur spécifiée dans la politique. Cette clé accepte également le nombre de secondes depuis le 1er janvier 1970.

- Availability (Disponibilité) : cette clé figure toujours dans le contexte de la demande.
- Type de données – Date, Numerique
- Type de valeur — Single-valued

aws:referer

Utilisez cette clé pour comparer le référent de la demande dans le navigateur client avec le référent spécifié dans la politique. La valeur `aws:referer` du contexte de la demande est fournie par le principal dans un en-tête HTTP. L'en-tête `Referer` est inclus dans une demande de navigateur Web lorsque vous sélectionnez un lien sur une page Web. L'en-tête `Referer` contient l'URL de la page Web où le lien a été sélectionné.

- Disponibilité : cette clé est incluse dans le contexte de la demande uniquement si la demande à la AWS ressource a été invoquée par un lien à partir de l'URL d'une page Web dans le navigateur.

Cette clé n'est pas incluse pour les demandes de programmation, car elle n'utilise pas de lien de navigateur pour accéder à la ressource AWS .

- Types de données – Chaîne
- Type de valeur — Single-valued

Par exemple, vous pouvez accéder à un objet Amazon S3 directement à l'aide d'une URL ou en utilisant l'invocation directe de l'API. Pour plus d'informations, consultez la section Opérations d'API Amazon S3 directement à l'aide d'un navigateur web. Lorsque vous accédez à un objet Amazon S3 à partir d'une URL qui existe dans une page web, l'URL de la page web source est utilisée dans `aws:referer`. Lorsque vous accédez à un objet Amazon S3 en saisissant l'URL dans votre navigateur, `aws:referer` n'est pas présent. Lorsque vous appelez l'API directement, `aws:referer` n'est pas non plus présent. Vous pouvez utiliser la clé de condition `aws:referer` dans une politique pour autoriser les demandes effectuées à partir d'un référent spécifique, comme un lien sur une page Web dans le domaine de votre entreprise.

> **Warning**
>
> Utilisez cette clé avec précaution. Il est dangereux d'inclure une valeur d'en-tête de référent connue publiquement. Les tiers non autorisés peuvent utiliser des navigateurs modifiés ou personnalisés pour fournir n'importe quelle valeur `aws:referer` de leur choix. Par conséquent, `aws:referer` il ne doit pas être utilisé pour empêcher des parties non autorisées de faire des AWS demandes directes. Cette clé est fournie uniquement pour permettre aux clients de protéger leur contenu numérique, stocké notamment dans Amazon S3, contre tout référencement sur des sites tiers non autorisés.

aws : RequestedRegion

Utilisez cette clé pour comparer la AWS région qui a été appelée dans la demande avec la région que vous spécifiez dans la politique. Vous pouvez utiliser cette clé de condition globale pour contrôler les régions qui peuvent être demandées. Pour afficher les AWS régions de chaque service, consultez la section Points de terminaison et quotas de service dans le Référence générale d'Amazon Web Services.

- Availability (Disponibilité) : cette clé figure toujours dans le contexte de la demande.
- Types de données – Chaîne
- Type de valeur — Single-valued

Certains services globaux, tels qu'IAM, ont un point de terminaison unique. Comme ce point de terminaison se trouve physiquement dans la région USA Est (Virginie du Nord), les appels IAM sont toujours effectués vers la région us-east-1. Par exemple, si vous créez une politique qui refuse l'accès à tous les services si la région demandée n'est pas us-west-2, les appels IAM échouent toujours. Pour voir un exemple de la façon de contourner ce problème, voir NotAction avec Refuser.

> **Note**
>
> La clé de condition `aws:RequestedRegion` vous permet de contrôler le point de terminaison de service qui est appelé, mais n'a pas d'impact sur l'opération. Certains services ont des impacts entre régions.
> Par exemple, Amazon S3 possède des opérations d'API qui s'étendent entre les régions.
>
> - Vous pouvez appeler `s3:PutBucketReplication` dans une région (qui est affectée par la clé de condition `aws:RequestedRegion`), mais d'autres régions sont affectées en fonction des paramètres de configuration des réplications.
> - Vous pouvez invoquer `s3:CreateBucket` pour créer un compartiment dans une autre région et utiliser la clé de condition `s3:LocationConstraint` pour contrôler les régions applicables.

Vous pouvez utiliser cette clé contextuelle pour limiter l'accès aux AWS services au sein d'un ensemble donné de régions. Par exemple, la politique suivante permet à un utilisateur d'afficher toutes les instances Amazon EC2 dans Console de gestion AWS. Cependant, elle lui permet d'apporter des modifications aux instances dans Irlande (eu-west-1), Londres (eu-west-2) ou Paris (eu-west-3).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "InstanceConsoleReadOnly",
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:Export*",
                "ec2:Get*",
```

```
                "ec2:Search*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "InstanceWriteRegionRestricted",
            "Effect": "Allow",
            "Action": [
                "ec2:Associate*",
                "ec2:Import*",
                "ec2:Modify*",
                "ec2:Monitor*",
                "ec2:Reset*",
                "ec2:Run*",
                "ec2:Start*",
                "ec2:Stop*",
                "ec2:Terminate*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestedRegion": [
                        "eu-west-1",
                        "eu-west-2",
                        "eu-west-3"
                    ]
                }
            }
        }
    ]
}
```

les lois :RequestTag/clé-balise

Utilisez cette clé pour comparer la paire clé-valeur de balise qui a été transmise dans la demande avec la paire de balises spécifiée dans la politique. Par exemple, vous pouvez vérifier que la demande comprend la clé de balise `"Dept"` et qu'elle a la valeur `"Accounting"`. Pour de plus amples informations, veuillez consulter Contrôle de l'accès lors AWS des demandes.

- Disponibilité : cette clé figure dans le contexte de la demande lorsque les balises sont transmises dans la demande. Lorsque plusieurs balises sont transmises dans la demande, il y a une clé de contexte pour chaque paire clé-valeur de balise.

- Types de données – Chaîne
- Type de valeur — Single-valued

Cette clé de contexte est formatée `"aws:RequestTag/`*`tag-key`*`":"`*`tag-value`*`"` où *`tag-key`* et *`tag-value`* constituent une paire clé de balise et valeur. Les clés de balises ne sont pas sensibles à la casse. Cela signifie que si vous spécifiez `"aws:RequestTag/TagKey1": "Value1"` dans l'élément de condition de votre politique, la condition correspond à une clé de balise de demande nommée `TagKey1` ou `tagkey1`, mais pas aux deux. Les valeurs de ces key/value paires de balises distinguent les majuscules et minuscules. Cela signifie que si vous spécifiez `"aws:RequestTag/TagKey1": "Production"` dans l'élément de condition de votre politique, la condition correspond à une clé de balise de demande nommée `Production`, mais elle ne correspond pas à `production` ou `PRODUCTION`.

Cet exemple montre que même si la clé est à valeur unique, vous pouvez toujours utiliser plusieurs paires clé-valeur dans une requête si les clés sont différentes.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "ec2:CreateTags",
    "Resource": "arn:aws:ec2::111122223333:instance/*",
    "Condition": {
      "StringEquals": {
        "aws:RequestTag/environment": [
          "preprod",
          "production"
        ],
        "aws:RequestTag/team": [
          "engineering"
        ]
      }
    }
  }
}
```

aws : TagKeys

Utilisez cette clé pour comparer les clés de balise d'une demande avec celles spécifiées dans la politique. Nous vous recommandons, lorsque vous utilisez des politiques pour contrôler l'accès à l'aide de balises, d'utiliser la clé de condition `aws:TagKeys` pour définir quelles clés de balises sont autorisées. Pour obtenir des exemples de stratégie et de plus amples informations, veuillez consulter the section called "Contrôle de l'accès en fonction des clés de balise".

- Disponibilité : cette clé figure dans le contexte de la demande si l'opération prend en charge le passage de balises dans la demande.
- Type de données – Chaîne (liste)
- Type de valeur – À valeur multiple

Cette clé de contexte est mise en forme à l'`'"aws:TagKeys":"`*`tag-key`*`"`endroit *`tag-key`* d'une liste de clés de balises sans valeurs (par exemple,`["Dept","Cost-Center"]`).

Étant donné que vous pouvez inclure plusieurs paires clé-valeur de balise dans une demande, le contenu de la demande peut être une demande à valeurs multiples. Dans ce cas, vous devez utiliser le opérateurs d'ensemble `ForAllValues` ou `ForAnyValue`. Pour de plus amples informations, veuillez consulter Définir les opérateurs pour les clés de contexte à valeurs multiples.

Certains services prennent en charge le balisage avec des opérations de ressource, comme la création, la modification ou de la suppression d'une ressource. Pour autoriser le balisage et les opérations sous la forme d'un seul appel, vous devez créer une politique qui inclut à la fois l'action de balisage et celle de modification de ressource. Vous pouvez ensuite utiliser la clé de condition `aws:TagKeys` pour imposer l'utilisation de clés de balise spécifiques dans la demande. Par exemple, pour limiter les balises lorsque quelqu'un crée un instantané Amazon EC2, vous devez inclure l'action de création `ec2:CreateSnapshot` et l'action de balisage `ec2:CreateTags` dans la politique. Pour afficher une politique correspondant à ce scénario, qui utilise `aws:TagKeys`, consultez Création d'un instantané avec des balises dans le Guide de l'utilisateur Amazon EC2.

aws : SecureTransport

Utilisez cette clé pour vérifier si une demande est envoyée avec TLS. Le contexte de la demande renvoie `true` ou `false`. Dans une politique, vous pouvez autoriser des actions spécifiques uniquement si la demande est envoyée avec TLS.

- Availability (Disponibilité) : cette clé figure toujours dans le contexte de la demande.

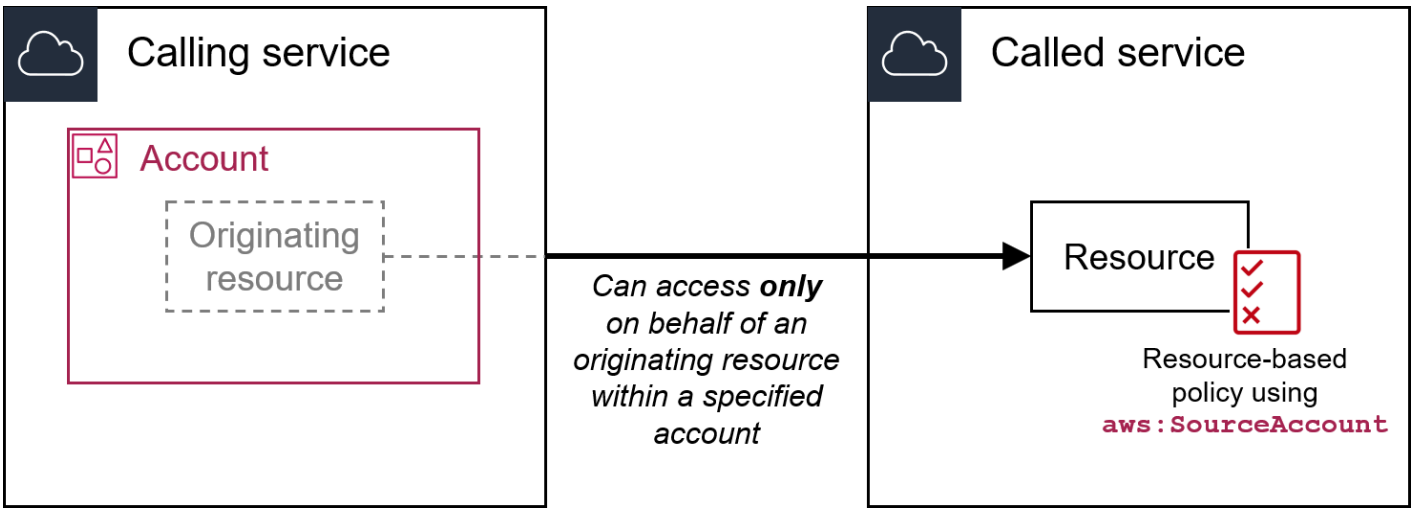
- Types de données – Booléen
- Type de valeur — Single-valued

aws : SourceAccount

Utilisez cette clé pour comparer l'ID de compte de la ressource effectuant une demande de service à service avec l'ID de compte que vous spécifiez dans la politique, mais uniquement lorsque la demande est faite par un AWS principal de service.

- Disponibilité : cette clé est incluse dans le contexte de la demande uniquement lorsque l'appel à votre ressource est effectué directement par un principal du service AWS au nom d'une ressource pour laquelle la configuration a déclenché la demande de service à service. Le service appelant transmet l'ID de compte de la ressource au service appelé.

> **Note**
>
> Cette clé fournit un mécanisme uniforme pour l'application du contrôle de l'adjoint confus entre les services au sein des Services AWS. Cependant, toutes les intégrations de services ne nécessitent pas l'utilisation de cette clé de condition globale. Consultez la documentation Services AWS que vous utilisez pour plus d'informations sur les mécanismes spécifiques aux services visant à atténuer les risques de confusion entre les services pour les adjoints.



- Types de données – Chaîne
- Type de valeur — Single-valued

Vous pouvez utiliser cette clé de condition pour vous assurer qu'un service d'appel peut accéder à votre ressource uniquement lorsque la demande provient d'un compte spécifique. Par exemple, vous pouvez attacher la politique de contrôle des ressources (RCP) suivante pour refuser les requêtes des principaux de service contre les compartiments Amazon S3, à moins qu'elles n'aient été déclenchées par une ressource du compte spécifié. Cette politique n'applique le contrôle que sur les requêtes des principaux de service (`"Bool": {"aws:PrincipalIsAWSService": "true"}`) qui ont la clé `aws:SourceAccount` présente (`"Null": {"aws:SourceAccount": "false"}`), de sorte que les intégrations de services qui ne nécessitent pas l'utilisation de cette clé et les appels de vos principaux ne sont pas impactés. Si la clé `aws:SourceAccount` est présente dans le contexte de la requête, la condition `Null` aura la valeur `true`, ce qui entraînera l'application de la clé `aws:SourceAccount`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RCPEnforceConfusedDeputyProtection",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceAccount": "111122223333"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
        }
      }
    }
  ]
}
```

Dans les politiques basées sur les ressources où le principal est un Service AWS principal, utilisez la clé pour limiter les autorisations accordées au service. Par exemple, lorsqu'un compartiment Amazon S3 est configuré pour envoyer des notifications à une rubrique Amazon SNS, le service Amazon S3 invoque l'opération d'API `sns:Publish` pour tous les événements configurés. Dans la politique de rubrique qui autorise l'opération `sns:Publish`, définissez la valeur de la clé de condition sur l'ID de compte du compartiment Amazon S3.

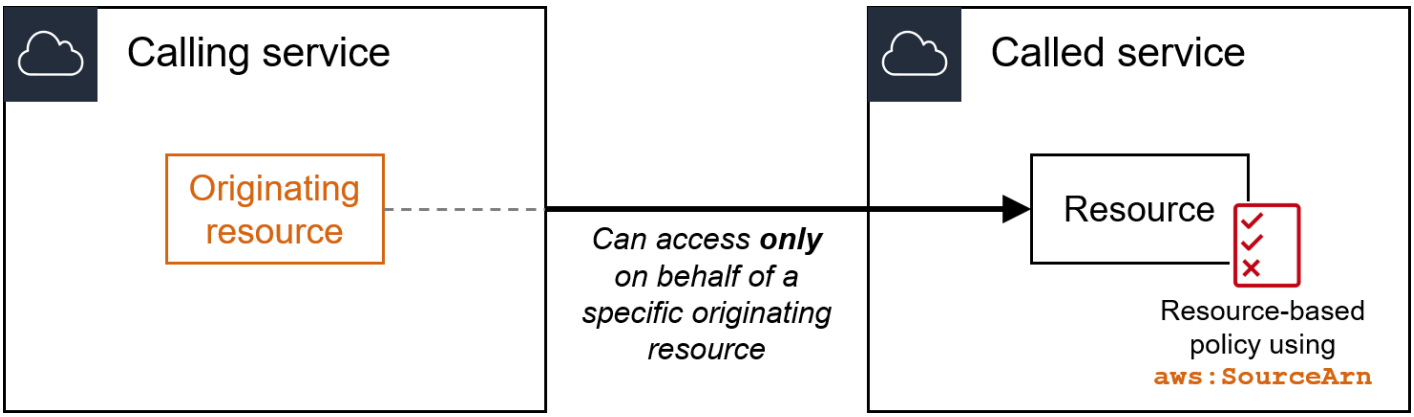
aws : SourceArn

Utilisez cette clé pour comparer l'Amazon Resource Name (ARN) de la ressource effectuant une demande de service à service avec l'ARN que vous spécifiez dans la politique, mais seulement lorsque la demande est faite par un principal du service AWS . Lorsque l'ARN de la source inclut l'ID de compte, il n'est pas nécessaire d'utiliser `aws:SourceAccount` avec `aws:SourceArn`.

Cette clé ne fonctionne pas avec l'ARN du principal qui fait la demande. Utilisez à la place `aws : PrincipalArn`.

- Disponibilité : cette clé est incluse dans le contexte de la demande uniquement lorsque l'appel à votre ressource est effectué directement par un principal du service AWS au nom d'une ressource pour laquelle la configuration a déclenché la demande de service à service. Le service appelant transmet l'ARN de la ressource d'origine au service appelé.

> **Note**
>
> Cette clé fournit un mécanisme uniforme pour l'application du contrôle de l'adjoint confus entre les services au sein des Services AWS. Cependant, toutes les intégrations de services ne nécessitent pas l'utilisation de cette clé de condition globale. Consultez la documentation Services AWS que vous utilisez pour plus d'informations sur les mécanismes spécifiques aux services visant à atténuer les risques de confusion entre les services pour les adjoints.



- Type de données : ARN

  AWS recommande d'utiliser des opérateurs ARN plutôt que des opérateurs de chaîne lors de la comparaison des ARN.

- Type de valeur — Single-valued

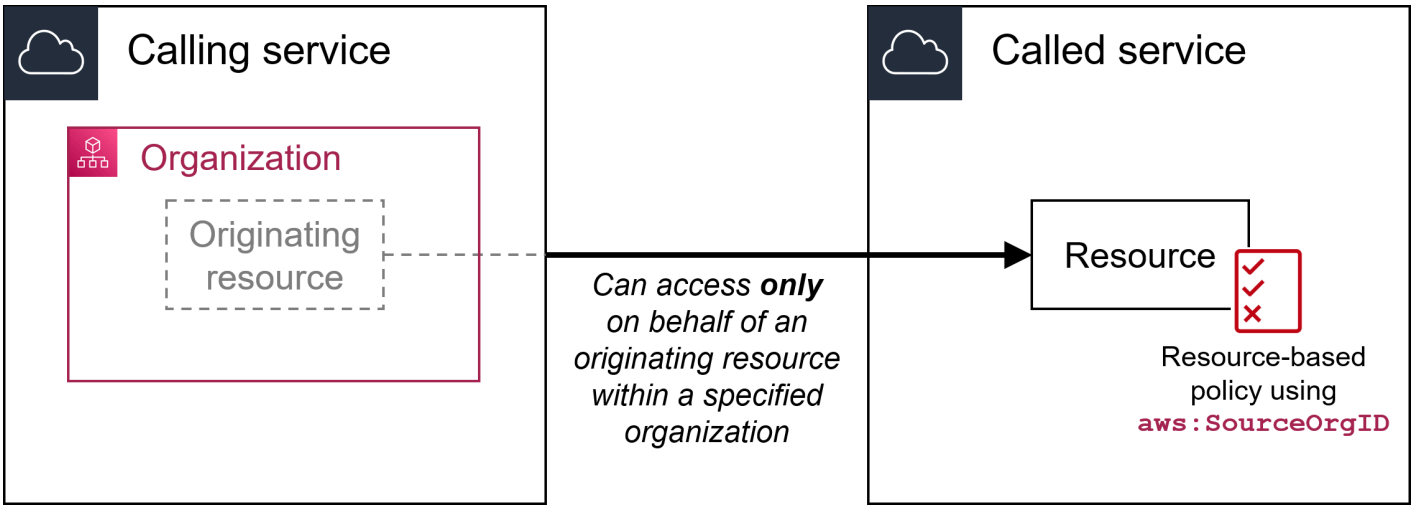
Vous pouvez utiliser cette clé de condition pour vous assurer qu'un service d'appel peut accéder à votre ressource uniquement lorsque la demande provient d'une ressource spécifique. Lorsque vous utilisez une politique basée sur les ressources avec un Service AWS principal comme principal`Principal`, définissez la valeur de cette clé de condition sur l'ARN de la ressource à laquelle vous souhaitez restreindre l'accès. Par exemple, lorsqu'un compartiment Amazon S3 est configuré pour envoyer des notifications à une rubrique Amazon SNS, le service Amazon S3 invoque l'opération d'API `sns:Publish` pour tous les événements configurés. Dans la politique de rubrique qui autorise l'opération `sns:Publish`, définissez la valeur de la clé de condition sur l'ARN du compartiment Amazon S3. Pour savoir quand utiliser cette clé de condition dans les politiques basées sur les ressources, consultez la documentation relative aux Services AWS que vous utilisez.

aws : SourceOrg ID

Utilisez cette clé pour comparer l'ID d'organisation de la ressource effectuant une demande de service à service avec l'ID d'organisation que vous spécifiez dans la politique, mais uniquement lorsque la demande est faite par un AWS responsable de service. Lorsque vous ajoutez et supprimez des comptes à une organisation dans AWS Organizations, les politiques qui contiennent la clé `aws:SourceOrgID` incluent automatiquement les bons comptes et vous n'avez pas besoin de mettre à jour manuellement les politiques.

- Disponibilité : cette clé est contenue dans le contexte de la demande uniquement lorsque l'appel à votre ressource est effectué directement par un principal du service AWS au nom d'une ressource appartenant à un compte membre d'une organisation. Le service appelant transmet l'ID de l'organisation de la ressource d'origine au service appelé.

> **Note**
>
> Cette clé fournit un mécanisme uniforme pour l'application du contrôle de l'adjoint confus entre les services au sein des Services AWS. Cependant, toutes les intégrations de services ne nécessitent pas l'utilisation de cette clé de condition globale. Consultez la documentation Services AWS que vous utilisez pour plus d'informations sur les mécanismes spécifiques aux services visant à atténuer les risques de confusion entre les services pour les adjoints.



- Types de données – Chaîne
- Type de valeur — Single-valued

Vous pouvez utiliser cette clé de condition pour vous assurer qu'un service d'appel peut accéder à votre ressource uniquement lorsque la demande provient d'une organisation spécifique. Par exemple, vous pouvez joindre la politique de contrôle des ressources (RCP) suivante pour refuser les demandes des responsables de service concernant des compartiments Amazon S3, sauf si elles ont été déclenchées par une ressource de l'organisation spécifiée. AWS Cette politique n'applique le contrôle que sur les requêtes des principaux de service (`"Bool": {"aws:PrincipalIsAWSService": "true"}`) qui ont la clé `aws:SourceAccount` présente

(`"Null": {"aws:SourceAccount": "false"}`), de sorte que les intégrations de services qui ne nécessitent pas l'utilisation de la clé et les appels de vos principaux ne sont pas impactés. Si la clé `aws:SourceAccount` est présente dans le contexte de la requête, la condition `Null` aura la valeur `true`, ce qui entraînera l'application de la clé `aws:SourceOrgID`. Nous utilisons `aws:SourceAccount` au lieu de `aws:SourceOrgID` dans l'opérateur de condition `Null` afin que le contrôle s'applique toujours si la requête provient d'un compte qui n'appartient pas à une organisation.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RCPEnforceConfusedDeputyProtection",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceOrgID": "o-xxxxxxxxxx"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
        }
      }
    }
  ]
}
```

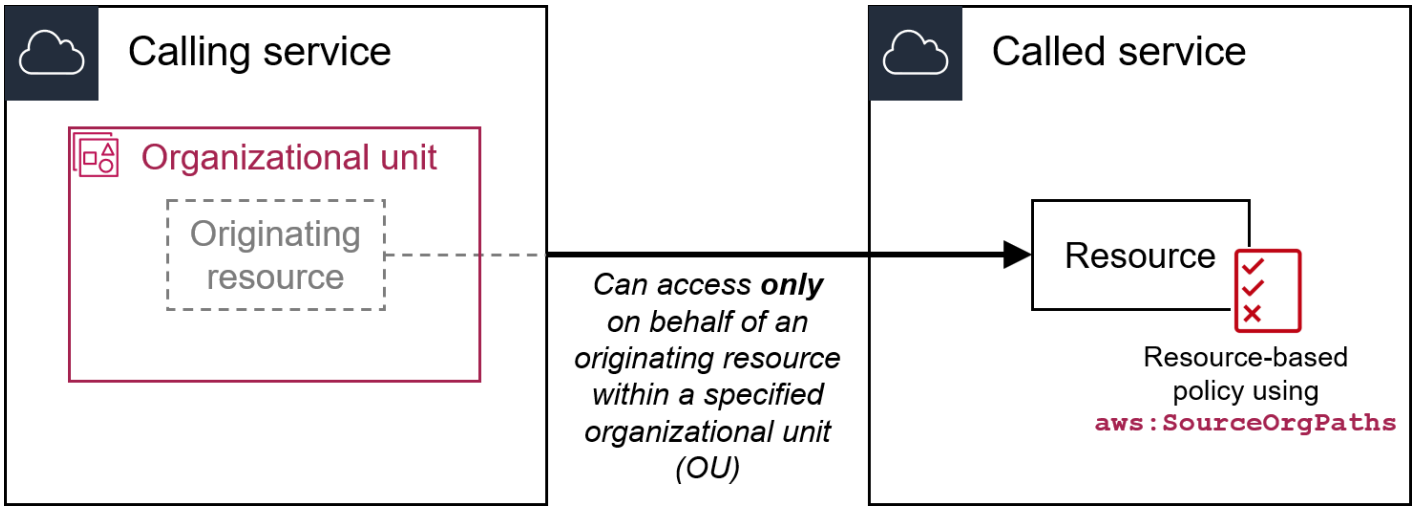
aws : SourceOrgPaths

Utilisez cette clé pour comparer le AWS Organizations chemin de la ressource effectuant une demande de service à service avec le chemin de l'organisation que vous spécifiez dans la

politique, mais uniquement lorsque la demande est faite par un AWS responsable de service. Un AWS Organizations chemin est une représentation textuelle de la structure d'une AWS Organizations entité. Pour de plus amples informations sur l'utilisation et la compréhension des chemins, veuillez consulter Présentation du chemin d'entité AWS Organizations.

- Disponibilité : cette clé est contenue dans le contexte de la demande uniquement lorsque l'appel à votre ressource est effectué directement par un principal du service AWS au nom d'une ressource appartenant à un compte membre d'une organisation. Le service appelant doit transmet le chemin de l'organisation de la ressource d'origine au service appelé.

> **Note**
>
> Cette clé fournit un mécanisme uniforme pour l'application du contrôle de l'adjoint confus entre les services au sein des Services AWS. Cependant, toutes les intégrations de services ne nécessitent pas l'utilisation de cette clé de condition globale. Consultez la documentation Services AWS que vous utilisez pour plus d'informations sur les mécanismes spécifiques aux services visant à atténuer les risques de confusion entre les services pour les adjoints.



- Type de données – Chaîne (liste)
- Type de valeur – À valeur multiple

Utilisez cette clé de condition pour vous assurer qu'un service d'appel peut accéder à votre ressource uniquement lorsque la demande provient d'une unité organisationnelle (UO) spécifique de AWS Organizations.

De la même manière que pour `aws:SourceOrgID`, afin d'éviter tout impact sur les intégrations de services qui ne nécessitent pas l'utilisation de cette clé, utilisez l'opérateur de condition `Null` avec la clé de condition `aws:SourceAccount` afin que le contrôle s'applique toujours si la requête provient d'un compte qui n'appartient pas à une organisation.

```
{
      "Condition": {
        "ForAllValues:StringNotLikeIfExists": {
            "aws:SourceOrgPaths": "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-
ab12-22222222/"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
        }
      }
}
```

`aws:SourceOrgPaths` est une clé de condition à valeurs multiples. Les clés multivaluées peuvent avoir plusieurs valeurs dans le contexte de la demande. Vous devez utiliser les opérateurs d'ensemble `ForAnyValue` ou `ForAllValues` avec l'opérateur de condition de chaîne pour cette clé. Pour de plus amples informations sur les clés de condition à valeurs multiples, veuillez consulter Définir les opérateurs pour les clés de contexte à valeurs multiples.

aws : UserAgent

Utilisez cette clé pour comparer l'application cliente du demandeur avec l'application spécifiée dans la politique.

- Availability (Disponibilité) : cette clé figure toujours dans le contexte de la demande.
- Types de données – Chaîne
- Type de valeur — Single-valued

> **Warning**
>
> Utilisez cette clé avec précaution. Dans la mesure où le principal fournit la valeur `aws:UserAgent` dans un en-tête HTTP, les tiers non autorisés peuvent modifier ou personnaliser les navigateurs de manière à fournir n'importe quelle valeur `aws:UserAgent`. Par conséquent, `aws:UserAgent` il ne doit pas être utilisé pour empêcher des parties non autorisées de faire des AWS demandes directes. Vous pouvez l'utiliser pour autoriser uniquement les applications clientes spécifiques, et uniquement après avoir testé votre politique.

aws : IsMcpServiceAction

Utilisez cette clé pour vérifier que l'action autorisée est une action du service MCP. Cette clé ne fait pas référence aux actions entreprises par le service MCP sur d'autres AWS services.

- Disponibilité : cette clé est incluse dans le contexte de la demande et définie sur True uniquement lorsque le service MCP autorise une action de service MCP.
- Types de données – Booléen
- Type de valeur — Single-valued

Autres clés de condition inter-services

AWS STS prend en charge les clés de condition de SAML-based fédération et les clés de condition interservices pour la fédération OIDC. Ces clés sont disponibles lorsqu'un utilisateur fédéré à l'aide d'OIDC ou de SAML effectue des AWS opérations dans d'autres services.

IAM et AWS STS clés de contexte de condition

Vous pouvez utiliser `Condition` cet élément dans une politique JSON pour tester la valeur des clés incluses dans le contexte de toutes les AWS demandes. Ces clés fournissent des informations sur la demande elle-même ou les ressources référencées par cette dernière. Vous pouvez vérifier que les clés disposent de valeurs spécifiées avant d'autoriser l'action demandée par l'utilisateur. Vous bénéficiez ainsi d'un contrôle détaillé sur la correspondance ou non de vos instructions de politique JSON avec une demande entrante. Pour plus d'informations sur l'utilisation de l'élément `Condition` dans une politique JSON, consultez Éléments de la politique JSON IAM : Condition.

Cette rubrique décrit les clés définies et fournies par le service IAM (avec un `iam:` préfixe) et le service AWS Security Token Service (AWS STS) (avec un `sts:` préfixe). Plusieurs autres AWS services fournissent également des clés spécifiques au service qui sont pertinentes pour les actions et les ressources définies par ce service. Pour plus d'informations, consultez la section Actions, ressources et clés de condition pour les AWS services. La documentation pour un service qui prend en charge des clés de condition possède généralement des informations supplémentaires. Par exemple, pour obtenir des informations sur les clés que pouvez utiliser dans les politiques relatives aux ressources Amazon S3, veuillez consulter clés de politique Amazon S3 dans le guide de l'utilisateur service de stockage simple Amazon.

Rubriques

- Clés disponibles pour IAM
- Clés disponibles pour AWS Fédération OIDC
- Clés disponibles pour SAML-based AWS STS fédération
- Cross-service SAML-based AWS STS clés de contexte de fédération
- Clés disponibles pour AWS STS

Clés disponibles pour IAM

Vous pouvez utilisez les clés de condition suivantes dans des politiques qui contrôlent l'accès aux ressources IAM :

iam : AccountPropertyNamespaces

Fonctionne avec des opérateurs de chaîne.

Spécifie les espaces de noms des propriétés du compte qui sont lus ou modifiés dans la demande. Les propriétés des comptes sont organisées en espaces de noms (par exemple,`RoleManager`). Lorsque vous appelez `PutAccountProperties` ou`GetAccountProperties`, utilisez cette clé de condition pour limiter les espaces de noms de propriétés auxquels un principal peut accéder.

iam : AssociatedResourceArn

Fonctionne avec des opérateurs ARN.

Spécifie l'ARN de la ressource à laquelle ce rôle sera associé dans le service de destination. La ressource appartient généralement au service auquel le principal transmet le rôle. Parfois,

la ressource peut appartenir à un troisième service. Par exemple, vous pouvez transmettre à Amazon EC2 Auto Scaling un rôle qui sera utilisé sur une instance Amazon EC2. Dans ce cas, la condition correspondrait à l'ARN de l'instance Amazon EC2.

Cette clé de condition s'applique uniquement à l'PassRoleaction d'une politique. Elle ne peut pas être utilisée pour limiter une autre action.

> **Important**
>
> Lorsque vous utilisez la `iam:AssociatedResourceArn` condition dans une politique pour restreindre l'PassRoleaction, des considérations particulières s'appliquent si la stratégie vise à définir l'accès à l'AddRoleToInstanceProfileaction. Dans ce cas, vous ne pouvez pas spécifier une région ou un identifiant d'instance dans l'ARN de l'instance EC2. La valeur de l'ARN doit être `arn:aws:ec2:*:CallerAccountId:instance/*`. L'utilisation de toute autre valeur ARN peut entraîner des résultats d'évaluation inattendus.

Utilisez cette clé de condition dans une politique basée sur l'identité pour permettre à une entité de transmettre un rôle, mais uniquement si ce rôle est associé à la ressource spécifiée. Par exemple, vous pouvez autoriser un utilisateur ou un rôle IAM à transmettre n'importe quel rôle au service Amazon EC2 à utiliser avec des instances dans le Compte AWS. L'utilisateur ou le rôle IAM ne serait pas autorisé à transmettre des rôles à d'autres services.

```
{
    "Effect": "Allow",
    "Action": "iam:PassRole",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "iam:PassedToService": "ec2.amazonaws.com"
        },
        "ArnLike": {
            "iam:AssociatedResourceARN": [
                "arn:aws:ec2:*:111122223333:instance/*"
            ]
        }
    }
}
```

> **Note**
>
> AWS services prenant en charge iam : prennent PassedToService également en charge cette clé de condition.

iam : AWSServiceName

Fonctionne avec des opérateurs de chaîne.

Spécifie le AWS service auquel ce rôle est attaché.

Cette clé de condition est prise en charge par l'opération de l'API `CreateServiceLinkedRole`.

> **Tip**
>
> Pour plus d'informations sur les services qui prennent en charge l'utilisation de rôles liés à des services, consultez AWS services qui fonctionnent avec IAM et recherchez les services pour lesquels la mention Oui apparaît dans la colonne Service-Linked Rôle. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Dans cet exemple, vous autorisez une entité à créer un rôle lié à un service en utilisant l'opération de l'API `CreateServiceLinkedRole` si le nom de service est access-analyzer.amazonaws.com.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
       "Effect": "Allow",
       "Action": "iam:CreateServiceLinkedRole",
       "Resource": "*",
       "Condition": {
         "StringLike": {
           "iam:AWSServiceName": "access-analyzer.amazonaws.com"
         }
       }
```

```
    }]
}
```

iam : FIDO-certification

Fonctionne avec des opérateurs de chaîne.

Vérifie le niveau de certification FIDO de l'appareil MFA au moment de l'enregistrement d'une clé de sécurité FIDO. La certification de l'appareil est extraite du FIDO Alliance Metadata Service (MDS). Si le statut ou le niveau de certification de votre clé de sécurité FIDO change, celui-ci ne sera pas mis à jour à moins que l'appareil ne soit désenregistré et enregistré de nouveau pour récupérer les informations de certification mises à jour.

Valeurs possibles de L1, L1plus, L2, L2plus, L3, L3plus

Dans cet exemple, vous enregistrez une clé de sécurité et récupérez la certification FIDO de niveau 1 plus pour votre appareil.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-certification": "L1plus"
                }
            }
        }
```

```
    ]

}
```

iam : FIDO-FIPS-140-2-certification

Fonctionne avec des opérateurs de chaîne.

Vérifie le niveau de certification de FIPS-140-2 validation de l'appareil MFA au moment de l'enregistrement d'une clé de sécurité FIDO. La certification de l'appareil est extraite du FIDO Alliance Metadata Service (MDS). Si le statut ou le niveau de certification de votre clé de sécurité FIDO change, celui-ci ne sera pas mis à jour à moins que l'appareil ne soit désenregistré et enregistré de nouveau pour récupérer les informations de certification mises à jour.

Valeurs possibles de L1, L2, L3, L4

Dans cet exemple, vous enregistrez une clé de sécurité et récupérez la certification de FIPS-140-2 niveau 2 de votre appareil.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
```

```
        }
    ]

}
```

iam : FIDO-FIPS-140-3-certification

Fonctionne avec des opérateurs de chaîne.

Vérifie le niveau de certification de FIPS-140-3 validation de l'appareil MFA au moment de l'enregistrement d'une clé de sécurité FIDO. La certification de l'appareil est extraite du FIDO Alliance Metadata Service (MDS). Si le statut ou le niveau de certification de votre clé de sécurité FIDO change, celui-ci ne sera pas mis à jour à moins que l'appareil ne soit désenregistré et enregistré de nouveau pour récupérer les informations de certification mises à jour.

Valeurs possibles de L1, L2, L3, L4

Dans cet exemple, vous enregistrez une clé de sécurité et récupérez la certification de FIPS-140-3 niveau 3 de votre appareil.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L3"
                }
```

```
            }
        }
    ]

}
```

iam : OrganizationsPolicyId

Fonctionne avec des opérateurs de chaîne.

Vérifie que la politique avec l' AWS Organizations ID spécifié correspond à la politique utilisée dans la demande. Pour voir un exemple de politique IAM qui utilise cette clé de condition, veuillez consulter IAM : Afficher les informations du dernier accès au service pour une politique AWS Organizations.

iam : PassedToService

Fonctionne avec des opérateurs de chaîne.

Spécifie le principal du service auquel un rôle peut être transmis. Cette clé de condition s'applique uniquement à l'PassRoleaction d'une politique. Elle ne peut pas être utilisée pour limiter une autre action.

Lorsque vous utilisez cette clé de condition dans une politique, spécifiez le service à l'aide d'un principal service. Un principal de service est le nom d'un service qui peut être spécifié dans l'élément `Principal` d'une politique. Il s'agit du format habituel :`SERVICE_NAME_URL.amazonaws.com`.

Vous pouvez utiliser `iam:PassedToService` pour limiter vos utilisateurs de sorte qu'ils puissent transmettre des rôles uniquement à des services spécifiques. Par exemple, un utilisateur peut créer un rôle de service qui approuve CloudWatch l'écriture des données de journal dans un compartiment Amazon S3 en son nom. Ensuite, l'utilisateur doit attacher une politique d'autorisation et une politique d'approbation au nouveau rôle de service. Dans ce cas, la politique d'approbation doit spécifier `cloudwatch.amazonaws.com` dans l'élément `Principal`. Pour consulter une politique qui permet à l'utilisateur de transmettre le rôle à CloudWatch, consultezIAM : transmettre un rôle IAM à un service spécifique AWS.

À l'aide de cette clé de condition, vous pouvez vous assurer que les utilisateurs créent des rôles de service uniquement pour les services que vous spécifiez. Par exemple, si un utilisateur de la politique précédente tente de créer un rôle de service pour Amazon EC2, l'opération échouera. L'échec se produit, car l'utilisateur n'est pas autorisé à transmettre le rôle à Amazon EC2.

Parfois, vous transmettez un rôle à un service, qui le transmet ensuite à un service différent. `iam:PassedToService` inclut uniquement le service final qui endosse le rôle, et non le service intermédiaire qui transmet le rôle.

> **Note**
>
> Certains services ne prennent pas en charge cette clé de condition.

iam : PermissionsBoundary

Fonctionne avec des opérateurs ARN.

Vérifie que la politique spécifiée est attachée en tant que limite d'autorisations sur la ressource du principal IAM. Pour plus d'informations, veuillez consulter Limites d'autorisations pour les entités IAM

iam:PolicyARN

Fonctionne avec des opérateurs ARN.

Vérifie l'ARN (Amazon Resource Name) d'une politique gérée dans les demandes qui impliquent ce type de politique. Pour de plus amples informations, veuillez consulter Contrôle de l'accès aux politiques.

Nom : ARNA RoleTemplate

Fonctionne avec des opérateurs ARN.

Spécifie l'ARN du modèle de rôle utilisé pour créer ou gérer un rôle. Lorsqu'un AWS service acquiert un rôle à partir d'un modèle de rôle, l'ARN du modèle de rôle est inclus en tant que clé de contexte dans les vérifications d'autorisation pour `CreateRole` `TagRole` `PutRolePolicyAttachRolePolicy`,,`PutRolePermissionsBoundary`, et`GetRole`.

iam : RegisterSecurityKey

Fonctionne avec des opérateurs de chaîne.

Vérifie l'état actuel de l'activation de l'appareil MFA.

Valeurs possibles de `Create` ou `Activate`.

Dans cet exemple, vous enregistrez une clé de sécurité et récupérez la certification de FIPS-140-3 niveau 1 pour votre appareil.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L1"
                }
            }
        }
    ]

 }
```

objectif :/ResourceTag*key-name*

Fonctionne avec des opérateurs de chaîne.

Vérifie que l'étiquette attachée à la ressource d'identité (utilisateur ou rôle) correspond aux nom et valeur de la clé spécifiée.

> **Note**
>
> IAM et AWS STS support à la fois pour la clé de condition `iam:ResourceTag` IAM et pour la clé de condition `aws:ResourceTag` globale.

Vous pouvez ajouter des attributs personnalisés à des ressources IAM sous la forme d'une paire clé-valeur. Pour plus d'informations sur les balises pour les ressources IAM, consultez the section called "Balises pour les ressources IAM". Vous pouvez utiliser `ResourceTag` pour contrôler l'accès aux ressources AWS , y compris les ressources IAM. Cependant, comme IAM ne prend pas en charge les balises pour les groupes, vous ne pouvez pas utiliser de balises pour contrôler l'accès aux groupes.

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise la suppression des utilisateurs avec la balise **`status=terminated`**. Pour utiliser cette politique, remplacez celle de *`italicized placeholder text`* l'exemple de politique par vos propres informations. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:DeleteUser",
        "Resource": "*",
        "Condition": {"StringEquals": {"iam:ResourceTag/status":
 "terminated"}}
    }]
}
```

iam : ServiceSpecificCredentialAgeDays

Fonctionne avec des opérateurs numériques.

Cette clé de condition limite la création d'informations d'identification spécifiques au service en fonction de leurs paramètres d'expiration. Elle vous permet de contrôler la durée maximale, en jours, des informations d'identification spécifiques au service qui peuvent être créées.

La plage valide pour les jours est comprise entre 1 et 36 600 (minimum 1 jour, maximum 36 600 jours).

Cette clé de condition est prise en charge par l'opération de l'API `CreateServiceSpecificCredential`.

Dans cet exemple, vous autorisez un utilisateur à créer des informations d'identification spécifiques au service Amazon Bedrock uniquement si elles expirent dans les 90 jours.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceSpecificCredential",
            "Resource": "arn:aws:iam::111122223333:user/username",
            "Condition": {
                "StringEquals": {
                    "iam:ServiceSpecificCredentialServiceName":
 "bedrock.amazonaws.com"
                },
                "NumericLessThanEquals": {
                    "iam:ServiceSpecificCredentialAgeDays": "90"
                }
            }
        }
    ]
}
```

iam : ServiceSpecificCredentialServiceName

Fonctionne avec des opérateurs de chaîne.

Spécifie quels AWS services peuvent être utilisés lors de la gestion des informations d'identification spécifiques à un service. Cette clé de condition vous permet de restreindre les AWS services autorisés lors de la gestion des informations d'identification spécifiques à un service.

Cette clé de condition est prise en charge par les opérations d'API suivante :

- CreateServiceSpecificCredential
- DeleteServiceSpecificCredential
- ResetServiceSpecificCredential
- UpdateServiceSpecificCredential

Les services suivants sont pris en charge pour les informations d'identification spécifiques au service avec leur formatage de valeur exact :

- `bedrock.amazonaws.com`

- `cassandra.amazonaws.com`
- `codecommit.amazonaws.com`

Dans cet exemple, vous autorisez un utilisateur à créer des informations d'identification spécifiques au service à l'aide de l'opération d'API `CreateServiceSpecificCredential` uniquement pour le service Amazon Bedrock.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceSpecificCredential",
            "Resource": "arn:aws:iam::111122223333:user/username",
            "Condition": {
                "StringEquals": {
                    "iam:ServiceSpecificCredentialServiceName":
 "bedrock.amazonaws.com"
                }
            }
        }
    ]
}
```

iam : DelegationDuration

Fonctionne avec des opérateurs numériques.

Filtre l'accès en fonction de la durée de l'accès temporaire demandé dans une demande de délégation.

Les fournisseurs de produits peuvent utiliser cette clé de condition pour contrôler la durée maximale qu'ils autorisent dans les demandes de délégation envoyées aux clients. La durée est spécifiée en secondes et détermine la durée de validité des informations d'identification temporaires une fois que le client a libéré le jeton d'échange. Cela aide les fournisseurs de produits à appliquer des politiques internes concernant les limites de durée d'accès en fonction de leur cas d'utilisation.

Cette clé de condition est prise en charge par l'opération de l'API `CreateDelegationRequest`.

Dans cet exemple, vous n'autorisez la création de demandes de délégation que si la durée demandée est inférieure ou égale à 7 200 secondes (2 heures).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateDelegationRequest",
            "Resource": "*",
            "Condition": {
                "NumericLessThanEquals": {
                    "iam:DelegationDuration": "7200"
                }
            }
        }
    ]
}
```

iam : NotificationChannel

Fonctionne avec des opérateurs ARN.

Filtre l'accès en fonction de l'ARN de rubrique Amazon SNS spécifié pour la réception des notifications de demande de délégation.

Les fournisseurs de produits peuvent utiliser cette clé de condition pour limiter les sujets SNS qui peuvent être utilisés pour les notifications de demande de délégation lors de l'appel CreateDelegationRequest d'API. Les fournisseurs de produits doivent spécifier un sujet SNS pour recevoir les notifications de changement d'état et échanger des jetons. Cela garantit que les notifications ne sont envoyées qu'aux canaux approuvés au sein de l'organisation du fournisseur du produit.

Cette clé de condition est prise en charge par l'opération de l'API `CreateDelegationRequest`.

Dans cet exemple, vous n'autorisez la création de demandes de délégation que si elles utilisent une rubrique SNS spécifique pour les notifications.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": "iam:CreateDelegationRequest",
            "Resource": "*",
            "Condition": {
                "ArnEquals": {
                    "iam:NotificationChannel": "arn:aws:sns:us-
east-1:123456789012:delegation-notifications"
                }
            }
        }
    ]
}
```

iam : TemplateArn

Fonctionne avec des opérateurs ARN.

Filtre l'accès en fonction du modèle de politique ARN utilisé pour définir les autorisations dans une demande de délégation.

Les fournisseurs de produits peuvent utiliser cette clé de condition pour contrôler quels modèles de politique peuvent être utilisés dans l'appel CreateDelegationRequest d'API. Les modèles de politique définissent les autorisations temporaires que les fournisseurs de produits demandent dans les comptes clients. Cela permet aux fournisseurs de produits de limiter les modèles de politiques enregistrés qui peuvent être utilisés lors de la création de demandes de délégation.

Cette clé de condition est prise en charge par l'opération de l'API `CreateDelegationRequest`.

Dans cet exemple, vous n'autorisez la création de demandes de délégation que si elles utilisent un modèle de politique provenant d'un domaine partenaire spécifique.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateDelegationRequest",
            "Resource": "*",
            "Condition": {
                "ArnLike": {
                    "iam:TemplateArn": "arn:aws:iam::*:delegation-template/
partner_*"
```

```
                }
            }
        }
    ]
}
```

iam : DelegationRequestOwner

Fonctionne avec des opérateurs ARN.

Filtre l'accès en fonction de l' AWS identité ou du principal propriétaire de la demande de délégation.

Les clients peuvent utiliser cette clé de condition pour contrôler qui peut effectuer des actions sur les demandes de délégation en fonction de la propriété. Le propriétaire d'une demande de délégation est l' AWS identité ou le responsable du compte client qui a initié ou reçu la demande de délégation.

Cette clé de condition est prise en charge par les opérations d'API suivante :

- `GetDelegationRequest`
- `AcceptDelegationRequest`
- `RejectDelegationRequest`
- `SendDelegationToken`
- `ListDelegationRequests`
- `UpdateDelegationRequest`

Dans cet exemple, vous autorisez les utilisateurs à gérer uniquement les demandes de délégation dont ils sont propriétaires.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetDelegationRequest",
                "iam:AcceptDelegationRequest",
                "iam:RejectDelegationRequest",
                "iam:SendDelegationToken",
```

```
                "iam:UpdateDelegationRequest",
                "iam:ListDelegationRequests"
            ],
            "Resource": "*",
            "Condition": {
                "ArnEquals": {
                    "iam:DelegationRequestOwner": "${aws:PrincipalArn}"
                }
            }
        }
    ]
}
```

Clés disponibles pour AWS Fédération OIDC

Vous pouvez utiliser la fédération OIDC pour fournir des informations de sécurité temporaires aux utilisateurs qui ont été authentifiés via un fournisseur d'identité (IdP) compatible OpenID Connect auprès d'un fournisseur d'identité IAM OpenID Connect (OIDC) de votre compte. AWS Parmi ces fournisseurs GitHub, citons Amazon Cognito, Login with Amazon et Google. Les jetons d'identité et d'accès de votre propre IdP peuvent être utilisés, ainsi que les jetons de compte de service accordés aux charges de travail Amazon Elastic Kubernetes Service.

Vous pouvez utiliser les clés contextuelles de condition AWS OIDC pour rédiger des politiques qui limitent l'accès des principaux fédérés aux ressources associées à un fournisseur, une application ou un utilisateur spécifique. Ces clés sont généralement utilisées dans la politique d'approbation d'un rôle. Définissez les clés de condition en utilisant le nom du fournisseur OIDC (`token.actions.githubusercontent.com`) suivi d'une demande (`:aud`) : **`token.actions.githubusercontent.com:aud`**.

Certaines clés de condition de fédération OIDC peuvent être utilisées dans la session de rôle pour autoriser l'accès aux ressources. Si la valeur est Oui dans la colonne Disponible en session, vous pouvez utiliser ces clés de condition dans les politiques pour définir les accès des utilisateurs dans d'autres AWS services. Lorsqu'une réclamation n'est pas disponible en session, la clé de contexte de condition OIDC ne peut être utilisée que dans une politique de confiance des rôles pour l'AssumeRoleWithWebIdentityauthentification initiale.

Sélectionnez votre IdP pour voir comment les demandes de votre IdP sont mappées aux clés de contexte de condition IAM dans AWS. Vous trouverez plus d'informations sur les touches pour GitHub et Google dans l'onglet Par défaut.

Default

Par défaut répertorie les revendications OIDC standard et la manière dont elles correspondent aux clés de contexte AWS STS conditionnelles dans AWS. Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés de condition AWS STS aux valeurs de la colonne de demandes JWT IdP. Utilisez ce mappage si votre IdP n'est pas répertorié dans les options de l'onglet.

GitHub Les flux de travail Actions et Google en sont quelques exemples IdPs qui utilisent l'implémentation par défaut dans leur jeton d'identification OIDC JWT.

AWS STS clé de condition	Demande JWT IdP	Disponible dans la session
amr	amr	Oui
aud	azp Si aucune valeur n'est définie pour `azp`, la clé de condition `aud` est mappée à la demande `aud`.	Oui
e-mail	e-mail	Non
oaud	aud	Non
sub	sub	Oui

Pour plus d'informations sur l'utilisation des clés contextuelles de condition OIDC avec GitHub, consultezConfiguration d'un rôle pour le fournisseur d'identité GitHub OIDC. Pour de plus amples informations sur les champs Google `aud` et `azp`, veuillez consulter le guide Google Identity Platform OpenID Connect.

amr

Fonctionne avec des opérateurs de chaîne. La clé comporte plusieurs valeurs, ce qui signifie que vous l'analysez dans une politique à l'aide d'opérateurs de définition de condition.

Exemple : `token.actions.githubusercontent.com:amr`

La référence des méthodes d'authentification inclut les informations de connexion de l'utilisateur. Elle contient les valeurs suivantes :

- Si l'utilisateur n'est pas authentifié, la clé contient uniquement `unauthenticated`.
- Si l'utilisateur est authentifié, la clé contient la valeur `authenticated` et le nom du fournisseur de connexions utilisé dans l'appel (`accounts.google.com`).

aud

Fonctionne avec des opérateurs de chaîne.

Exemples :

- `accounts.google.com:aud`
- `token.actions.githubusercontent.com:aud`

Utilisez la clé de condition `aud` pour vérifier que le public correspond à celui spécifié dans la politique. Vous pouvez utiliser la clé aud avec la clé sub pour le même fournisseur d'identité.

Cette clé de condition est définie à partir des champs de jeton suivants :

- `aud` pour les ID de client Google OAuth 2.0 de votre application, lorsque le champ `azp` n'est pas défini. Lorsque le champ `azp` est défini, le champ `aud` correspond à la clé de condition `accounts.google.com:oaud`.
- `azp` lorsque le champ `azp` est défini. Cela peut se produire pour les applications hybrides au sein desquelles une application web et une application Android ont un ID de client Google OAuth 2.0 différent mais partagent le même projet d'API Google.

Lorsque vous écrivez une politique à l'aide de la clé de condition `accounts.google.com:aud`, vous devez savoir si l'application est une application hybride qui définit le champ `azp`.

Champ non défini `azp`

L'exemple de politique suivant fonctionne pour les applications non hybrides qui ne définissent pas le champ `azp`. Dans ce cas, la valeur du champ de jeton d'ID Google `aud` correspond à la fois aux valeurs de clé de condition `accounts.google.com:aud` et `accounts.google.com:oaud`.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "accounts.google.com"},
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "aud-value",
                    "accounts.google.com:oaud": "aud-value",
                    "accounts.google.com:sub": "sub-value"
                }
            }
        }
    ]
}
```

Champ défini `azp`

L'exemple de politique suivant fonctionne pour les applications hybrides qui définissent le champ `azp`. Dans ce cas, la valeur du champ de jeton d'ID Google `aud` correspond uniquement à la valeur de clé de condition `accounts.google.com:oaud`. La valeur du champ `azp` correspond à la valeur de clé de condition `accounts.google.com:aud`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "accounts.google.com"},
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "azp-value",
                    "accounts.google.com:oaud": "aud-value",
                    "accounts.google.com:sub": "sub-value"
                }
            }
        }
    ]
}
```

email

Fonctionne avec des opérateurs de chaîne.

Exemple : `accounts.google.com:email`

Cette clé de condition valide l'adresse e-mail de l'utilisateur. La valeur de cette réclamation n'est peut-être pas propre à ce compte et peut changer au fil du temps. Par conséquent, vous ne devez pas utiliser cette valeur comme identifiant principal pour vérifier votre enregistrement d'utilisateur.

oaud

Fonctionne avec des opérateurs de chaîne.

Exemple : `accounts.google.com:oaud`

Cette clé spécifie l'autre public (aud) auquel ce jeton d'ID est destiné. Il doit s'agir de l'un des ID client OAuth 2.0 de votre application.

sub

Fonctionne avec des opérateurs de chaîne.

Exemples :

- `accounts.google.com:sub`
- token.actions.githubusercontent.com:sub

Utilisez ces clés pour vérifier que le sujet correspond à celui spécifié dans la politique. Vous pouvez utiliser la clé `sub` avec la clé `aud` pour le même fournisseur d'identité.

Dans la politique de confiance des rôles suivante, la clé de `sub` condition limite le rôle à la GitHub branche nomméedemo.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
```

```
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
token.actions.githubusercontent.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
          "token.actions.githubusercontent.com:sub": "repo:org-name/repo-
name:ref:refs/heads/demo"
        }
      }
    }
  ]
}
```

Amazon Cognito

Cet onglet explique comment Amazon Cognito mappe les revendications OIDC pour AWS STS conditionner les clés de contexte. AWS Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés de condition AWS STS aux valeurs de la colonne de demandes JWT IdP.

Pour les rôles utilisés par Amazon Cognito, les clés sont définies par l'URL `cognito-identity.amazonaws.com` suivie de la demande.

Pour plus d'informations sur le mappage des demandes relatives au groupe d'identités, consultez la section Mappages des fournisseurs par défaut dans le Guide du développeur Amazon Cognito. Pour plus d'informations sur le mappage des demandes relatives au groupe d'utilisateurs, consultez la section Utilisation du jeton d'identification dans le Guide du développeur Amazon Cognito.

AWS STS clé de condition	Demande JWT IdP	Disponible dans la session
amr	amr	Oui
aud	aud	Oui
oaud	aud	Non
sub	sub	Oui

amr

Fonctionne avec des opérateurs de chaîne. La clé comporte plusieurs valeurs, ce qui signifie que vous l'analysez dans une politique à l'aide d'opérateurs de définition de condition.

Exemple : `cognito-identity.amazonaws.com:amr`

La référence des méthodes d'authentification inclut les informations de connexion de l'utilisateur. Elle contient les valeurs suivantes :

- Si l'utilisateur n'est pas authentifié, la clé contient uniquement `unauthenticated`.
- Si l'utilisateur est authentifié, la clé contient la valeur `authenticated` et le nom du fournisseur de connexions utilisé dans l'appel (`cognito-identity.amazonaws.com`).

Par exemple, la condition suivante de la politique d'approbation d'un rôle Amazon Cognito détermine si l'utilisateur n'est pas authentifié.

```
"Condition": {
  "StringEquals":
    { "cognito-identity.amazonaws.com:aud": "us-east-2:identity-pool-id" },
  "ForAnyValue:StringLike":
    { "cognito-identity.amazonaws.com:amr": "unauthenticated" }
}
```

aud

Fonctionne avec des opérateurs de chaîne.

Exemple : `cognito-identity.amazonaws.com:aud`

Client d'application du groupe d'utilisateurs qui a authentifié l'utilisateur. Amazon Cognito affiche la même valeur dans le champ standard `client_id` du jeton d'accès.

oaud

Fonctionne avec des opérateurs de chaîne.

Exemple : `cognito-identity.amazonaws.com:oaud`

Client d'application du groupe d'utilisateurs qui a authentifié l'utilisateur. Amazon Cognito affiche la même valeur dans le champ standard `client_id` du jeton d'accès.

sub

Fonctionne avec des opérateurs de chaîne.

Exemple : `cognito-identity.amazonaws.com:sub`

Identifiant unique (UUID), ou sujet, de l'utilisateur authentifié. Le nom d'utilisateur n'est peut-être pas unique dans votre groupe d'utilisateurs. La sous-revendication est le meilleur moyen d'identifier un utilisateur donné. Vous pouvez utiliser la clé `sub` avec la clé `aud` pour le même fournisseur d'identité.

```
"Condition": {
         "StringEquals": {
            "cognito-identity.amazonaws.com:aud": "us-east-1:12345678-abcd-abcd-
abcd-123456790ab",
            "cognito-identity.amazonaws.com:sub": [
               "us-east-1:12345678-1234-1234-1234-123456790ab",
               "us-east-1:98765432-1234-1234-1243-123456790ab"
            ]
         }
      }
```

Login with Amazon

Cet onglet explique comment Login with Amazon maps OIDC prétend AWS STS conditionner les clés de contexte. AWS Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés de condition AWS STS aux valeurs de la colonne de demandes JWT IdP.

AWS STS clé de condition	Demande JWT IdP	Disponible dans la session
app_id	ID d'application	Oui
sub	ID de l'utilisateur	Oui
user_id	ID de l'utilisateur	Oui

app_id

Fonctionne avec des opérateurs de chaîne.

Exemple : `www.amazon.com:app_id`

Cette clé spécifie le contexte d'audience qui correspond au champ `aud` utilisé par d'autres fournisseurs d'identité.

sub

Fonctionne avec des opérateurs de chaîne.

Exemple : `www.amazon.com:sub`

Cette clé vérifie que l'ID utilisateur correspond à celui spécifié dans la politique. Vous pouvez utiliser la clé `sub` avec la clé `aud` pour le même fournisseur d'identité.

user_id

Fonctionne avec des opérateurs de chaîne.

Exemple : `www.amazon.com:user_id`

Cette clé spécifie le contexte d'audience qui correspond au champ `aud` utilisé par d'autres fournisseurs d'identité. Vous pouvez utiliser la clé `user_id` avec la clé `id` du même fournisseur d'identité.

Facebook

Cet onglet explique comment Facebook Maps prétend que l'OIDC AWS STS conditionne les clés contextuelles. AWS Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés de condition AWS STS aux valeurs de la colonne de demandes JWT IdP.

AWS STS clé de condition	Demande JWT IdP	Disponible dans la session
app_id	ID d'application	Oui
id	id	Oui

app_id

Fonctionne avec des opérateurs de chaîne.

Exemple : `graph.facebook.com:app_id`

Cette clé spécifie que le contexte d'audience correspond au champ `aud` utilisé par d'autres fournisseurs d'identité.

id

Fonctionne avec des opérateurs de chaîne.

Exemple : `graph.facebook.com:id`

Cette clé vérifie que l'ID de l'application (ou du site) correspond à celui spécifié dans la politique.

GitHub

Cet onglet explique comment GitHub Actions mappe les revendications OIDC pour AWS STS conditionner les clés de contexte. AWS Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés de condition AWS STS aux valeurs de la colonne de demandes JWT IdP.

AWS STS clé de condition	Demande JWT IdP	Disponible dans la session
acteur	acteur	Non
actor_id	actor_id	Non
ref_job_workflow_ref	ref_job_workflow_ref	Non
référentiels	référentiels	Non
id du référentiel	id du référentiel	Non
id du propriétaire du référentiel	id du propriétaire du référentiel	Non
flux de travail	flux de travail	Non
ref	ref	Non
environment	environment	Non
identifiant_entreprise	identifiant_entreprise	Non

acteur

Fonctionne avec des opérateurs de chaîne.

Exemple : `token.actions.githubusercontent.com:actor`

Cette clé identifie le compte personnel qui a lancé l'exécution du flux de travail. Utilisez-le pour restreindre l'accès à des acteurs spécifiques.

identifiant de l'acteur

Fonctionne avec des opérateurs de chaîne.

Exemple : `token.actions.githubusercontent.com:actor_id`

Cette clé permet de vérifier l'ID du compte personnel qui a lancé l'exécution du flux de travail. Les identifiants d'acteurs sont générés par GitHub et sont immuables.

ref_job_workflow_ref

Fonctionne avec des opérateurs de chaîne.

Exemple : `token.actions.githubusercontent.com:job_workflow_ref`

Cette clé contient le chemin de référence vers le flux de travail réutilisable pour les tâches utilisant un flux de travail réutilisable. Utilisez-le pour restreindre l'accès à des flux de travail spécifiques et pour vous assurer que seuls les flux de travail approuvés peuvent assumer des rôles.

référentiel

Fonctionne avec des opérateurs de chaîne.

Exemple : `token.actions.githubusercontent.com:repository`

Cette clé identifie le référentiel à partir duquel le flux de travail est exécuté. Utilisez-le pour limiter l'accès à des GitHub référentiels spécifiques.

id du référentiel

Fonctionne avec des opérateurs de chaîne.

Exemple : `token.actions.githubusercontent.com:repository_id`

Cette clé vérifie l'ID du référentiel à partir duquel le flux de travail est exécuté. Les ID de référentiel sont immuables et ne changent pas même si le référentiel est renommé.

id du propriétaire du référentiel

Fonctionne avec des opérateurs de chaîne.

Exemple : `token.actions.githubusercontent.com:repository_owner_id`

Cette clé permet de vérifier l'ID de l'organisation dans laquelle le référentiel est stocké. L'ID du propriétaire du référentiel est un identifiant unique et stable qui ne change pas.

flux de travail

Fonctionne avec des opérateurs de chaîne.

Exemple : `token.actions.githubusercontent.com:workflow`

Cette clé contient le nom du flux de travail. Utilisez-le pour restreindre l'accès à des flux de travail spécifiques au sein de vos référentiels.

réf

Fonctionne avec des opérateurs de chaîne.

Exemple : `token.actions.githubusercontent.com:ref`

Cette clé identifie la référence git (branche ou balise) qui a déclenché l'exécution du workflow. Utilisez-le pour limiter l'accès en fonction de branches spécifiques, par exemple autoriser uniquement `main` ou `production` branches.

environment

Fonctionne avec des opérateurs de chaîne.

Exemple : `token.actions.githubusercontent.com:environment`

Cette clé contient le nom de l'environnement utilisé par la tâche. Utilisez-le pour implémenter des contrôles d'accès basés sur l'environnement, tels que des autorisations distinctes pour les environnements de développement, de test et de production.

> **Note**
>
> Si la réclamation environnementale est incluse dans votre politique de confiance, un environnement doit être configuré et fourni dans le GitHub flux de travail.

identifiant_entreprise

Fonctionne avec des opérateurs de chaîne.

Exemple : `token.actions.githubusercontent.com:enterprise_id`

Cette clé vérifie l'ID de l'entreprise qui contient le référentiel à partir duquel le flux de travail est exécuté. Utilisez-le pour vous assurer que l'accès est limité aux référentiels de votre organisation GitHub d'entreprise.

⚠ Utilisez des identifiants immuables, pas des noms

Activé GitHub, les noms du référentiel, de l'organisation et des utilisateurs peuvent changer. Un nom libéré en le renommant ou en le supprimant peut être revendiqué par un autre compte. Les politiques qui s'appuient uniquement sur des revendications mutables basées sur des noms (telles que `repository` ou`actor`) peuvent autoriser l'accès à des identités involontaires. Pour vous protéger contre cela, utilisez les identifiants immuables qui GitHub fournissent, par exemple, ou `repository_id` `repository_owner_id`or`actor_id`, dans votre rôle des politiques de confiance ou des politiques de contrôle des ressources.

L'exemple de politique de confiance suivant utilise des revendications personnalisées dans le jeton GitHub OIDC pour limiter l'accès à un rôle.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "Federated": "arn:aws:iam::111122223333:oidc-provider/
token.actions.githubusercontent.com"
         },
         "Action": "sts:AssumeRoleWithWebIdentity",
         "Condition": {
            "StringEquals": {
                "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
                "token.actions.githubusercontent.com:repository_owner_id": "123456",
                "token.actions.githubusercontent.com:repository_id": "1296269",
```

```
                "token.actions.githubusercontent.com:actor_id": "1",
                "token.actions.githubusercontent.com:ref": "refs/heads/main",
                "token.actions.githubusercontent.com:enterprise_id": "345"
              }
          }
      }
  ]
}
```

Google

Cet onglet explique comment l'OIDC de Google Maps prétend AWS STS conditionner les clés contextuelles. AWS Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés de condition AWS STS aux valeurs de la colonne de demandes JWT IdP.

AWS STS clé de condition	Demande JWT IdP	Disponible dans la session
google/organization_number	google : numéro d'organisation	Non

google/organization_number

Fonctionne avec des opérateurs numériques.

Exemple : `accounts.google.com:google/organization_number`

Cette clé permet de vérifier qu'un jeton représente une identité Google appartenant à une organisation Google Cloud ou Google Workspace spécifique. Utilisez-le pour restreindre l'accès aux utilisateurs d'organisations spécifiques, en veillant à ce que seules les identités de votre organisation puissent assumer le rôle.

L'exemple de politique de confiance suivant utilise la `google/organization_number` revendication pour limiter l'accès à un rôle.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
```

```
            "Federated": "accounts.google.com"
         },
         "Action": "sts:AssumeRoleWithWebIdentity",
         "Condition": {
            "NumericEquals": {
                "accounts.google.com:google/organization_number": "123456"
               }
           }
        }
    ]
}
```

CircleCI

Cet onglet explique comment CircleCI mappe les revendications OIDC pour AWS STS conditionner les clés de contexte. AWS Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés de condition AWS STS aux valeurs de la colonne de demandes JWT IdP.

AWS STS clé de condition	Demande JWT IdP	Disponible dans la session
oidc.circleci. com/project-identifiant	oidc.circleci. com/project-identifiant	Non

oidc.circleci. com/project-identifiant

Fonctionne avec des opérateurs de chaîne.

Exemple : `circleci-issuer-url:oidc.circleci.com/project-id`

Cette clé permet d'identifier le projet CircleCI dans lequel la tâche est exécutée. Sa valeur est une chaîne contenant un UUID qui identifie de manière unique le projet CircleCI. Utilisez-le pour restreindre l'accès à des projets CircleCI spécifiques.

L'exemple de politique de confiance suivant utilise la `oidc.circleci.com/project-id` revendication pour limiter l'accès à un rôle.

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::123456789012:oidc-provider/oidc.circleci.com/
org/12345"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.circleci.com/org/12345:aud": "sts.amazonaws.com",
          "oidc.circleci.com/org/12345:oidc.circleci.com/project-id": "76543210-
ba98-fedc-3210-edcba0987654"
        }
      }
    }
  ]
}
```

Oracle Cloud Infrastructure (OCI)

Cet onglet explique comment Oracle Cloud Infrastructure mappe les revendications OIDC pour AWS STS conditionner les clés de contexte. AWS Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés de condition AWS STS aux valeurs de la colonne de demandes JWT IdP.

AWS STS clé de condition	Demande JWT IdP	Disponible dans la session
rpst_id	rpst_id	Non

rpst_id

Fonctionne avec des opérateurs de chaîne.

Exemple : `oci-issuer-url:rpst_id`

Cette clé identifie de manière unique le principal de la ressource dans OCI. Utilisez-le pour restreindre l'accès à des principaux de ressources OCI spécifiques. Le rpst_id (Resource Principal Session Token ID) fournit un identifiant stable pour l'authentification OCI basée sur les ressources.

L'exemple de politique de confiance suivant utilise la `rpst_id` revendication pour limiter l'accès à un rôle.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::123456789012:oidc-provider/idcs-
abc123ef5678901234abcd.identity.oraclecloud.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "idcs-abc123ef5678901234abcd.identity.oraclecloud.com:aud":
 "sts.amazonaws.com",
          "idcs-abc123ef5678901234abcd.identity.oraclecloud.com:rpst_id": "your-
rpst-id"
        }
      }
    }
  ]
}
```

GitLab.com

Cet onglet explique comment GitLab.com CI/CD mappe les affirmations de l'OIDC pour AWS STS conditionner les clés de contexte. AWS Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés de condition AWS STS aux valeurs de la colonne de demandes JWT IdP.

> **Note**
>
> Ces clés de condition personnalisées ne sont prises en charge que lors de l'utilisation de l'instance SaaS GitLab.com multi-tenant en tant que fournisseur d'identité OIDC. Vous ne pouvez pas utiliser ces clés de condition avec GitLab Dedicated ou avec GitLab Self-Managed des instances.

AWS STS clé de condition	Demande JWT IdP	Disponible dans la session
namespace_id	namespace_id	Non
project_id	project_id	Non
user_id	user_id	Non
identifiant_utilisateur	identifiant_utilisateur	Non
email_utilisateur	email_utilisateur	Non
niveau_d'accès utilisateur	niveau_d'accès utilisateur	Non
ref_protected	ref_protected	Non
source_du_pipeline	source_du_pipeline	Non
runner_environment	runner_environment	Non

namespace_id

Fonctionne avec des opérateurs de chaîne.

Exemple : `gitlab.com:namespace_id`

Cette clé identifie l'identifiant de l'espace de noms (groupe) du projet qui exécute la CI/CD tâche. L'ID de l'espace de noms est un identifiant unique et stable qui ne change pas si le groupe est renommé.

id_projet

Fonctionne avec des opérateurs de chaîne.

Exemple : `gitlab.com:project_id`

Cette clé identifie l'ID du projet qui exécute la CI/CD tâche. L'ID de projet est un identifiant unique et stable qui ne change pas si le projet est renommé ou déplacé.

user_id

Fonctionne avec des opérateurs de chaîne.

Exemple : `gitlab.com:user_id`

Cette clé identifie l'ID de l'utilisateur qui exécute la CI/CD tâche. L'ID utilisateur est un identifiant unique et stable.

identifiant_utilisateur

Fonctionne avec des opérateurs de chaîne.

Exemple : `gitlab.com:user_login`

Cette clé identifie le nom d'utilisateur de l'utilisateur qui exécute la CI/CD tâche. Les noms d'utilisateur peuvent être modifiés ; pour un identifiant stable, utilisez-le à la `user_id` place.

email_utilisateur

Fonctionne avec des opérateurs de chaîne.

Exemple : `gitlab.com:user_email`

Cette clé identifie l'adresse e-mail de l'utilisateur qui exécute la CI/CD tâche. La valeur de cette revendication peut changer au fil du temps ; pour un identifiant stable, utilisez-le à la `user_id` place.

niveau_d'accès utilisateur

Fonctionne avec des opérateurs de chaîne.

Exemple : `gitlab.com:user_access_level`

Cette clé identifie le niveau d'accès de l'utilisateur au sein du projet (par exemple,`maintainer`,`developer`,`owner`).

ref_protected

Fonctionne avec des opérateurs de chaîne.

Exemple : `gitlab.com:ref_protected`

Cette clé indique si la référence Git qui a déclenché la tâche est protégée. La valeur est « `true` si protégé », `false` sinon.

source_du_pipeline

Fonctionne avec des opérateurs de chaîne.

Exemple : `gitlab.com:pipeline_source`

Cette clé identifie la source qui a déclenché le pipeline (par exemple`push`,`web`,`schedule`,`api`,`merge_request_event`).

runner_environment

Fonctionne avec des opérateurs de chaîne.

Exemple : `gitlab.com:runner_environment`

Cette clé permet d'identifier le type de patin utilisé pour la CI/CD tâche.

> **Important**
>
> En GitLab.com mode SaaS, les chemins de groupes ou de projets supprimés pouvaient auparavant être recréés par différents utilisateurs, ce qui pouvait entraîner des politiques reposant uniquement sur des revendications basées sur des chemins (telles que`sub`) pour accorder l'accès à des identités involontaires. Au 1er juin 2026, GitLab a déployé une mise à jour côté plateforme qui empêche l'émission de nouveaux jetons d'identification CI pour les parcours de projet qui émettaient auparavant des jetons, réduisant ainsi ce risque à l'avenir. En tant que niveau de protection supplémentaire, AWS et GitLab recommandez d'inclure des conditions relatives aux identifiants stables et uniques, tels que `namespace_id` ou`project_id`, dans les politiques de confiance ou les politiques de contrôle des ressources de votre rôle.

L'exemple de politique de confiance suivant utilise la `namespace_id` revendication conjointement `sub` pour limiter l'accès à un GitLab.com projet et à une branche spécifiques.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::123456789012:oidc-provider/gitlab.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "gitlab.com:sub": "project_path:mygroup/
myproject:ref_type:branch:ref:main",
```

```
          "gitlab.com:namespace_id": "12345"
        }
      }
    }
  ]
}
```

Buildkite

Cet onglet explique comment Buildkite mappe les affirmations OIDC pour AWS STS conditionner les clés de contexte. AWS Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés de condition AWS STS aux valeurs de la colonne de demandes JWT IdP.

AWS STS clé de condition	Demande JWT IdP	Disponible dans la session
organisation_slug	organisation_slug	Non
organization_id	organization_id	Non
pipeline_slug	pipeline_slug	Non
identifiant du pipeline	identifiant du pipeline	Non
nom_cluster	nom_cluster	Non
identifiant_cluster	identifiant_cluster	Non
build_branch	build_branch	Non

organisation_slug

Fonctionne avec des opérateurs de chaîne.

Exemple : `agent.buildkite.com:organization_slug`

Cette clé identifie l'organisation Buildkite à partir de laquelle le job est exécuté. Utilisez-le pour restreindre l'accès à des organisations spécifiques.

identifiant de l'organisation

Fonctionne avec des opérateurs de chaîne.

Exemple : `agent.buildkite.com:organization_id`

Cette clé vérifie l'ID de l'organisation Buildkite à partir de laquelle le job est exécuté. Les identifiants d'organisation sont générés par Buildkite et sont immuables.

pipeline_slug

Fonctionne avec des opérateurs de chaîne.

Exemple : `agent.buildkite.com:pipeline_slug`

Cette clé permet d'identifier le pipeline dans lequel la tâche est exécutée. Utilisez-le pour restreindre l'accès à des pipelines Buildkite spécifiques.

identifiant du pipeline

Fonctionne avec des opérateurs de chaîne.

Exemple : `agent.buildkite.com:pipeline_id`

Cette clé permet de vérifier l'ID du pipeline dans lequel la tâche est exécutée. Les identifiants de pipeline sont générés par Buildkite et sont immuables.

nom_cluster

Fonctionne avec des opérateurs de chaîne.

Exemple : `agent.buildkite.com:cluster_name`

Cette clé identifie le cluster sur lequel la tâche est exécutée. Utilisez-le pour restreindre l'accès à des clusters Buildkite spécifiques.

identifiant_cluster

Fonctionne avec des opérateurs de chaîne.

Exemple : `agent.buildkite.com:cluster_id`

Cette clé permet de vérifier l'ID du cluster sur lequel la tâche est exécutée. Les identifiants de cluster sont générés par Buildkite et sont immuables.

build_branch

Fonctionne avec des opérateurs de chaîne.

Exemple : `agent.buildkite.com:build_branch`

Cette clé identifie la branche git qui a déclenché la compilation. Utilisez-le pour limiter l'accès en fonction de branches spécifiques, par exemple autoriser uniquement `main` ou `production` branches.

> ⚠ Utilisez des identifiants immuables, pas des slugs
>
> Sur Buildkite, une organisation ou un slug de pipeline supprimé peut être immédiatement réenregistré par une autre organisation ou un autre pipeline. Buildkite émet des jetons OIDC pour ces slugs auprès du même émetteur (). `agent.buildkite.com` Par conséquent, les politiques qui s'appuient uniquement sur des allégations basées sur des limaces (telles que `organization_slug` ou`pipeline_slug`) pourraient autoriser l'accès à des identités involontaires. Pour vous protéger contre cela, utilisez des identifiants stables et immuables, tels que `organization_id` ou, dans votre rôle`pipeline_id`, des politiques de confiance ou des politiques de contrôle des ressources.

L'exemple de politique de confiance suivant utilise des revendications personnalisées dans le jeton OIDC Buildkite pour limiter l'accès à un rôle.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "Federated": "arn:aws:iam::123456789012:oidc-provider/
agent.buildkite.com"
        },
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {
            "StringEquals": {
                "agent.buildkite.com:aud": "sts.amazonaws.com",
                "agent.buildkite.com:organization_id": "0191e7a0-1234-7abc-
def0-123456789abc",
                "agent.buildkite.com:pipeline_id": "0191e7a1-5678-7abc-
def0-123456789abc",
                "agent.buildkite.com:build_branch": "main"
```

```
            }
        }
    }
    ]
}
```

Plus d'informations sur la fédération OIDC

- Guide de l'utilisateur d'Amazon Cognito
- Fédération OIDC

Clés disponibles pour SAML-based AWS STS fédération

Si vous utilisez la SAML-based fédération à l'aide de AWS Security Token Service (AWS STS), vous pouvez inclure des clés de condition supplémentaires dans la politique.

Politiques d'approbation de rôle SAML

Dans la politique d'approbation d'un rôle, vous pouvez inclure les clés suivantes afin d'établir si le principal est autorisé à endosser le rôle. Hormis `saml:doc`, toutes les valeurs sont dérivées de l'assertion SAML. Tous les éléments de la liste sont disponibles dans l'éditeur visuel de la console IAM lorsque vous créez ou modifiez une politique avec des conditions. Les éléments marqués d'un `[]` peuvent avoir une valeur qui est une liste du type spécifié.

saml:aud

Fonctionne avec des opérateurs de chaîne.

L'URL d'un point de terminaison auquel les assertions SAML sont présentées. La valeur de cette clé provient du champ `SAML Recipient` de l'assertion, non du champ `Audience`.

saml:commonName[]

Fonctionne avec des opérateurs de chaîne.

Attribut `commonName`.

saml:cn[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduOrg`.

saml:doc

Fonctionne avec des opérateurs de chaîne.

Représente le principal utilisé pour endosser le rôle. Le format est *`account-ID`*/*`provider-friendly-name`*, par exemple`123456789012/SAMLProviderName`. La valeur ID compte fait référence au compte qui est propriétaire du fournisseur SAML.

saml:edupersonaffiliation[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonassurance[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonentitlement[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonnickname[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonorgdn

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonorgunitdn[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonprimaryaffiliation

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonprimaryorgunitdn

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonprincipalname

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonscopedaffiliation[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersontargetedid[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:eduorghomepageuri[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduOrg`.

saml:eduorgidentityauthnpolicyuri[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduOrg`.

saml:eduorglegalname[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduOrg`.

saml:eduorgsuperioruri[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduOrg`.

saml:eduorgwhitepagesuri[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduOrg`.

saml:givenName[]

Fonctionne avec des opérateurs de chaîne.

Attribut `givenName`.

saml:iss

Fonctionne avec des opérateurs de chaîne.

Auteur, représenté par un URN.

saml:mail[]

Fonctionne avec des opérateurs de chaîne.

Attribut `mail`.

saml:name[]

Fonctionne avec des opérateurs de chaîne.

Attribut `name`.

saml:namequalifier

Fonctionne avec des opérateurs de chaîne.

Valeur de hachage basée sur le nom convivial du fournisseur SAML. La valeur est la concaténation des valeurs suivantes, dans l'ordre et séparées par le caractère '/' :

1. Valeur de la réponse `Issuer` (`saml:iss`)

2. ID du compte AWS.
3. Nom convivial (dernière partie de l'ARN) du fournisseur SAML dans IAM

La concaténation de l'ID de compte et du nom convivial du fournisseur SAML est disponible dans les politiques IAM en tant que clé `saml:doc`. Pour plus d'informations, veuillez consulterIdentification unique des utilisateurs au sein de SAML-based la fédération.

saml:organizationStatus[]

Fonctionne avec des opérateurs de chaîne.

Attribut `organizationStatus`.

saml:primaryGroupSID[]

Fonctionne avec des opérateurs de chaîne.

Attribut `primaryGroupSID`.

saml:sub

Fonctionne avec des opérateurs de chaîne.

Objet de la demande, qui inclut une valeur qui identifie de manière unique un utilisateur individuel au sein d'une organisation (par exemple, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).

saml:sub_type

Fonctionne avec des opérateurs de chaîne.

Cette clé peut avoir la valeur `persistent`, `transient`, ou consister en l'URI `Format` complet des éléments `Subject` et `NameID` utilisés dans votre assertion SAML. La valeur `persistent` indique que la valeur de `saml:sub` reste la même pour l'utilisateur entre les sessions. Si la valeur est `transient`, l'utilisateur a une valeur `saml:sub` différente pour chaque session. Pour plus d'informations sur l'attribut `NameID` de l'élément `Format`, consultez Configuration d'assertions SAML pour la réponse d'authentification.

saml:surname[]

Fonctionne avec des opérateurs de chaîne.

Attribut `surnameuid`.

saml:uid[]

Fonctionne avec des opérateurs de chaîne.

Attribut `uid`.

échantillon : 500 [] UniqueIdentifier

Fonctionne avec des opérateurs de chaîne.

Attribut `x500UniqueIdentifier`.

Pour obtenir des informations générales sur les attributs `eduPerson` et `eduOrg`, accédez au site Web EFEDS Wiki. Pour obtenir la liste des attributs `eduPerson`, consultez Spécifications des classes d'objets eduPerson (201602).

Les clés de condition dont le type est une liste peuvent inclure plusieurs valeurs. Pour créer des conditions dans la politique pour des valeurs de liste, vous pouvez utiliser des opérateurs de définition (`ForAllValues`, `ForAnyValue`). Par exemple, pour autoriser un utilisateur dont l'affiliation est « faculté » ou « personnel » (mais pas « étudiant »), vous pouvez utiliser la condition suivante :

```
"Condition": {
   "ForAllValues:StringLike": {
     "saml:edupersonaffiliation":[ "faculty", "staff"]
   }
}
```

Cross-service SAML-based AWS STS clés de contexte de fédération

Certaines clés SAML-based de condition de fédération peuvent être utilisées dans des demandes ultérieures pour autoriser AWS des opérations dans d'autres services et `AssumeRole` appels. Les clés de condition suivantes peuvent être utilisées dans les politiques de confiance des rôles lorsque les principaux fédérés assument un autre rôle, et dans les politiques de ressources d'autres AWS services pour autoriser l'accès aux ressources par les principaux fédérés. Pour plus d'informations sur l'utilisation de ces clés, consultez la section À propos de la fédération SAML 2.0.

Sélectionnez une clé de condition pour voir la description.

- saml:namequalifier
- saml:sub

- saml:sub_type

> **Note**
>
> Aucune autre clé de condition de SAML-based fédération n'est disponible après la réponse d'authentification initiale du fournisseur d'identité externe (IdP).

Clés disponibles pour AWS STS

Vous pouvez utiliser les clés de condition suivantes dans les politiques de confiance des rôles IAM pour les rôles qui sont assumés à l'aide des opérations AWS Security Token Service (AWS STS).

saml:sub

Fonctionne avec des opérateurs de chaîne.

Objet de la demande, qui inclut une valeur qui identifie de manière unique un utilisateur individuel au sein d'une organisation (par exemple, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).

ensembles : AWSServiceName

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour spécifier un service dans lequel un jeton porteur fonctionne. Lorsque vous utilisez cette clé de condition dans une politique, spécifiez le service à l'aide d'un principal service. Un principal de service est le nom d'un service qui peut être spécifié dans l'élément `Principal` d'une politique. Par exemple, `codeartifact.amazonaws.com` c'est le directeur du AWS CodeArtifact service.

Disponibilité : cette clé est présente dans les demandes qui obtiennent un jeton porteur. Vous ne pouvez pas passer un appel direct AWS STS pour obtenir un jeton au porteur. Lorsque vous effectuez certaines opérations dans d'autres services, le service demande le jeton au porteur en votre nom.

Certains AWS services nécessitent que vous soyez autorisé à obtenir un jeton de support de AWS STS service avant de pouvoir accéder à leurs ressources par programmation. Par exemple, AWS CodeArtifact exige des principaux qu'ils utilisent des jetons porteurs pour effectuer certaines opérations. La commande `aws codeartifact get-authorization-token` renvoie un

jeton porteur. Vous pouvez ensuite utiliser le jeton porteur pour effectuer des AWS CodeArtifact opérations. Pour de plus amples informations sur les jetons porteurs, veuillez consulter Jeton porteur de service.

Vous pouvez utiliser cette clé de condition pour permettre aux entités d'obtenir un jeton porteur à utiliser avec un service spécifique.

ensembles : DurationSeconds

Fonctionne avec des opérateurs numériques.

Utilisez cette clé pour spécifier la durée (en secondes) qu'un mandant peut utiliser pour obtenir un jeton AWS AWS STS porteur ou un jeton Web JSON à partir de l'GetWebIdentityTokenAPI.

Disponibilité : cette clé est présente dans les demandes qui obtiennent un jeton porteur ou un jeton Web JSON à partir de l' GetWebIdentityToken API. Vous ne pouvez pas passer un appel direct AWS STS pour obtenir un jeton au porteur. Lorsque vous effectuez certaines opérations dans d'autres services, le service demande le jeton au porteur en votre nom. La clé n'est pas applicable aux opérations d' AWS STS attribution de rôle.

Certains AWS services nécessitent que vous soyez autorisé à obtenir un jeton de support de AWS STS service avant de pouvoir accéder à leurs ressources par programmation. Par exemple, AWS CodeArtifact exige des principaux qu'ils utilisent des jetons porteurs pour effectuer certaines opérations. La commande `aws codeartifact get-authorization-token` renvoie un jeton porteur. Vous pouvez ensuite utiliser le jeton porteur pour effectuer des AWS CodeArtifact opérations. Pour de plus amples informations sur les jetons porteurs, veuillez consulter Jeton porteur de service.

ensembles : IdentityTokenAudience

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour spécifier le public pour lequel un responsable IAM peut demander des jetons Web JSON (JWT) à l'aide de l'API. GetWebIdentityToken Lorsque cette clé de condition est présente dans une politique IAM, les responsables IAM ne peuvent demander des jetons que pour les audiences spécifiées dans la politique. Les services externes valident la demande d'audience (« aud ») dans le jeton Web JSON pour s'assurer que le jeton leur était destiné.

Disponibilité : cette clé est présente dans les requêtes adressées à l' GetWebIdentityToken API qui est utilisée pour obtenir des jetons Web JSON (JWT) à des fins d'authentification auprès de services externes.

Lorsque vous utilisez cette clé de condition dans une politique, spécifiez la valeur d'audience qui correspond à l'identifiant du destinataire prévu (par exemple,https://api.example.com).

L'exemple de politique suivant permet à un principal de demander des jetons pour les services externes spécifiés :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "sts:IdentityTokenAudience": [
                        "https://api2.example.com",
                        "https://api1.example.com"
                    ]
                }
            }
        }
    ]
}
```

ensembles : SigningAlgorithm

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour spécifier l'algorithme cryptographique AWS AWS STS utilisé pour signer les jetons Web JSON (JWT) générés par l'API. GetWebIdentityToken Lorsque vous utilisez cette clé de condition dans une politique, spécifiez ES384 (ECDSA avec P-384 courbe et SHA-384) ou RS256 (RSA avec). SHA-256

Disponibilité : cette clé est présente dans les requêtes adressées à l' GetWebIdentityToken API qui est utilisée pour obtenir des jetons Web JSON (JWT) à des fins d'authentification auprès de services externes.

Vous pouvez utiliser cette clé de condition pour faire en sorte que les responsables IAM demandent des jetons à l'aide d'algorithmes de signature compatibles avec vos exigences de sécurité ou avec les services externes auxquels vous intégrez. L'ES384 offre une sécurité et

des performances optimales, tandis que le RS256 offre une plus grande compatibilité avec les systèmes qui ne prennent pas en charge l'ECDSA.

L'exemple de politique suivant oblige les mandants à utiliser l'algorithme de signature ES384 :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "sts:SigningAlgorithm": "ES384"
                }
            }
        }
    ]
}
```

ensembles : ExternalId

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour exiger qu'un principal fournisse un identifiant spécifique lors de la prise en charge d'un rôle IAM.

Disponibilité : cette clé est présente dans la demande lorsque le principal fournit un identifiant externe tout en assumant un rôle à l'aide de l' AWS API AWS CLI or.

Identifiant unique qui peut être requis lorsque vous endossez un rôle dans un autre compte. Si l'administrateur du compte auquel appartient le rôle vous a fourni un ID externe, indiquez cette valeur dans le paramètre `ExternalId`. Cette valeur peut être n'importe quelle chaîne, comme une phrase de passe ou un numéro de compte. La fonction principale de l'ID externe est de traiter et de prévenir le problème du délégué confus. Pour de plus amples informations sur l'ID externe et le problème du député confus, veuillez consulter Accès à Comptes AWS appartenant à des tiers.

La valeur `ExternalId` peut avoir un minimum de 2 caractères et un maximum de 1 224 caractères. La valeur doit être alphanumérique sans espaces. Elle peut également inclure

les symboles suivants : signe plus (+), signe égal (=), virgule (,), point (.), arobase (@), deux points (:), barre oblique (/) et tiret (-).

sts :RequestContext//touche de contexte

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour comparer les paires clé-valeur du contexte de session intégrées dans l'assertion contextuelle signée par l'émetteur du jeton sécurisé transmise dans la demande avec les paires clé-valeurs spécifiées dans la politique de confiance des rôles.

Disponibilité : cette clé est présente dans la demande lorsqu'une assertion de contexte est fournie dans le paramètre de `ProvidedContexts` demande tout en assumant un rôle à l'aide de l'opération AWS STS AssumeRole d'API.

Cette clé de contexte a le format suivant : `"sts:RequestContext/context-key":"context-value"` où `context-key` et `context-value` constituent une paire clé-valeur de contexte. Lorsque plusieurs clés de contexte sont intégrées dans l'assertion de contexte signée transmise dans la demande, il y a une clé de contexte pour chaque paire clé-valeur. Vous devez autoriser l'action `sts:SetContext` dans la politique de confiance des rôles afin de permettre à un principal de définir des clés de contexte dans le jeton de session obtenu. Pour en savoir plus sur les clés de contexte IAM Identity Center prises en charge qui peuvent être utilisées avec cette clé, consultez la section clés de condition AWS STS pour IAM Identity Center dans le Guide de l'utilisateur AWS IAM Identity Center .

Vous pouvez utiliser cette clé dans une politique de confiance des rôles pour appliquer un contrôle d'accès précis basé sur l'utilisateur ou ses attributs lorsqu'il endosse un rôle. Une fois le rôle assumé, l'activité apparaît dans les AWS CloudTrail journaux de l'`AdditionalEventData`attribut, contenant les paires clé-valeur du contexte de session définies par le fournisseur de contexte dans la demande d'attribution du rôle. Cela permet aux administrateurs de différencier plus facilement les sessions de rôle lorsqu'un rôle est utilisé par différents principaux. Les paires clé-valeur sont définies par le fournisseur de contexte spécifié, et non par AWS CloudTrail ou. AWS STS Cela permet au fournisseur de contexte de contrôler le contexte inclus dans les CloudTrail journaux et les informations de session.

ensembles : RequestContextProviders

Fonctionne avec des opérateurs ARN.

Utilisez cette clé pour comparer l'ARN du fournisseur de contexte dans la demande avec l'ARN du fournisseur de contexte spécifié dans la politique de confiance des rôles.

Disponibilité : cette clé est présente dans la demande lorsqu'une assertion de contexte est fournie dans le paramètre de `ProvidedContexts` demande tout en assumant un rôle à l'aide de l'opération AWS STS AssumeRole d'API.

L'exemple de condition suivant vérifie que l'ARN du fournisseur de contexte transmis dans la demande correspond à l'ARN spécifié dans la condition de politique d'approbation des rôles. Nous vous recommandons d'ajouter une vérification de valeur nulle avec `ForAllValues` afin d'éviter que les clés de contexte manquantes ou les clés de contexte avec des valeurs vides ne soient évaluées comme vraies. Pour en savoir plus, consultez Opérateur de condition pour vérifier l'existence de clés de condition.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Action": "sts:SetContext",
    "Effect": "Allow",
    "Resource": "*",
    "Condition": {
      "ForAllValues:ArnEquals": {
        "sts:RequestContextProviders": [
          "arn:aws:iam::aws:contextProvider/IdentityCenter"
        ]
      },
      "Null": {
        "sts:RequestContextProviders": "false"
      }
    }
  }
}
```

ensembles : RoleAuthorizedByIdp

Fonctionne avec des opérateurs booléens.

Utilisez cette clé pour vérifier que le fournisseur d'identité (IdP) a explicitement autorisé le rôle demandé par le biais de la `https://aws.amazon.com/roles` réclamation figurant dans le jeton OIDC. La revendication de rôles contient les ARN des rôles IAM que le jeton est autorisé à assumer. AWS STS prend en charge deux formats pour la valeur de la réclamation :

Tableau JSON (recommandé) :

```
"https://aws.amazon.com/roles": [
  "arn:aws:iam::111122223333:role/RoleA",
  "arn:aws:iam::111122223333:role/RoleB"
]
```

Semicolon-delimited chaîne :

```
"https://aws.amazon.com/roles": "arn:aws:iam::111122223333:role/
RoleA;arn:aws:iam::111122223333:role/RoleB"
```

Disponibilité : cette clé est présente dans les requêtes qui utilisent cette `AssumeRoleWithWebIdentity` opération. La valeur est `true` lorsque le jeton inclut la `https://aws.amazon.com/roles` revendication et que l'ARN du rôle demandé correspond à une valeur de la revendication. La valeur correspond au `false` moment où le jeton n'inclut pas la réclamation. Comme la clé est toujours présente dans les `AssumeRoleWithWebIdentity` demandes, utilisez plutôt l'opérateur de `Bool` condition`BoolIfExists`.

Lorsque le jeton inclut la revendication mais que l'ARN du rôle demandé ne correspond pas à une valeur de la revendication, l'`AssumeRoleWithWebIdentity`appel renvoie une `InvalidIdentityToken` erreur.

Vous pouvez utiliser cette clé de condition dans les politiques de confiance des rôles, les politiques de contrôle des ressources (RCP) et les politiques de point de terminaison VPC pour exiger qu'un IdP autorise le rôle avant qu'il ne soit assumé.

La politique de confiance des rôles suivante `AssumeRoleWithWebIdentity` n'est autorisée que lorsque l'IdP a autorisé le rôle demandé dans le jeton.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
```

```
            "Condition": {
                "StringEquals": { "idp.example.com:aud": "my-app-id" },
                "Bool": { "sts:RoleAuthorizedByIdp": "true" }
            }
        }
    ]
}
```

ensembles : RoleSessionName

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour comparer le nom de session spécifié par un principal lorsqu'il endosse un rôle avec la valeur spécifiée dans la politique.

Disponibilité : cette clé est présente dans la demande lorsque le principal assume le rôle à l' Console de gestion AWS aide de toute commande CLI assume-role ou de toute opération d' AWS STS `AssumeRole`API.

Vous pouvez utiliser cette clé dans une politique d'approbation de rôle pour exiger que vos utilisateurs fournissent un nom de session spécifique lorsqu'ils endossent un rôle. Par exemple, vous pouvez exiger que les utilisateurs IAM spécifient leur propre nom d'utilisateur comme nom de session. Une fois que l'utilisateur IAM endosse le rôle, l'activité apparaît dans les journaux AWS CloudTrail sous le nom de session correspondant à son nom d'utilisateur. Cela permet aux administrateurs de différencier plus facilement les sessions de rôle lorsqu'un rôle est utilisé par différents principaux.

La politique d'approbation de rôle suivante exige que les utilisateurs IAM dans le compte `111122223333` fournissent leur nom d'utilisateur IAM comme nom de session lorsqu'ils endossent le rôle. Cette exigence est appliquée à l'aide de la variable de condition `aws:username` dans la clé de condition. Cette politique permet aux utilisateurs IAM d'endosser le rôle auquel la politique est attachée. Cette politique empêche quiconque utilise des informations d'identification temporaires d'endosser le rôle, car la variable `username` est uniquement présente pour les utilisateurs IAM.

> **Important**
>
> Vous pouvez utiliser n'importe quelle clé de condition à valeur unique comme variable. Vous ne pouvez pas utiliser de clé de condition à valeurs multiples en tant que variable.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "RoleTrustPolicyRequireUsernameForSessionName",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
            "Condition": {
                "StringLike": {"sts:RoleSessionName": "prefix-
${aws:username}"}
            }
        }
    ]
}
```

Lorsqu'un administrateur consulte le AWS CloudTrail journal d'une action, il peut comparer le nom de la session aux noms d'utilisateur de son compte. Dans l'exemple suivant, l'utilisateur appelé `matjac` a effectué l'opération en utilisant le rôle appelé `MateoRole`. L'administrateur peut alors contacter Mateo Jackson, dont le nom d'utilisateur est `matjac`.

```
    "assumedRoleUser": {
        "assumedRoleId": "AROACQRSTUVWRAOEXAMPLE:matjac",
        "arn": "arn:aws:sts::111122223333:assumed-role/MateoRole/matjac"
    }
```

Si vous autorisez l'accès entre comptes à l'aide de rôles, les utilisateurs d'un compte peuvent endosser un rôle dans un autre compte. L'ARN de l'utilisateur du rôle assumé répertorié dans CloudTrail inclut le compte où le rôle existe. Il n'inclut pas le compte de l'utilisateur qui a endossé le rôle. Les utilisateurs ne sont uniques qu'au sein d'un compte. Par conséquent, nous vous recommandons d'utiliser cette méthode pour consulter les CloudTrail journaux uniquement pour les rôles assumés par les utilisateurs dans les comptes que vous administrez. Vos utilisateurs peuvent utiliser le même nom d'utilisateur dans plusieurs comptes.

ensembles : SourceIdentity

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour comparer l'identité source spécifiée par un principal lorsqu'il endosse un rôle avec la valeur spécifiée dans la politique.

Disponibilité : cette clé est présente dans la demande lorsque le principal fournit une identité de source tout en assumant un rôle à l'aide d'une commande CLI AWS STS assume-role ou d'une opération d' AWS STS `AssumeRole`API.

Vous pouvez utiliser cette clé dans une politique d'approbation de rôle pour exiger que vos utilisateurs définissent une identité source spécifique lorsqu'ils endossent un rôle. Par exemple, vous pouvez exiger que votre main-d'œuvre ou vos identités fédérées spécifient une valeur pour l'identité source. Vous pouvez configurer votre fournisseur d'identité (IdP) de sorte qu'il utilise l'un des attributs associés à vos utilisateurs, un nom d'utilisateur ou un e-mail par exemple, comme identité source. L'IdP transmet ensuite l'identité de la source en tant qu'attribut dans les assertions ou les revendications qu'il envoie à. AWS La valeur de l'attribut d'identité source identifie l'utilisateur ou l'application qui endosse le rôle.

Une fois que l'utilisateur endosse le rôle, l'activité apparaît dans les journaux AWS CloudTrail avec la valeur d'identité source qui a été définie. Cela permet aux administrateurs de déterminer plus facilement qui a effectué des actions dans lesquelles un rôle a été attribué AWS. Vous devez octroyer des autorisations pour l'action `sts:SetSourceIdentity` afin d'autoriser une identité à définir une identité source.

Contrairement à `sts:RoleSessionName`, une fois l'identité source définie, la valeur ne peut plus être modifiée. Elle est présente dans le contexte de la demande pour toutes les actions effectuées avec le rôle par l'identité source. La valeur persiste dans les sessions de rôle suivantes lorsque vous utilisez les informations d'identification de session pour endosser un autre rôle. Le fait d'endosser un rôle à partir d'un autre est appelé chaînage des rôles.

Vous pouvez utiliser la clé de condition `aws:SourceIdentity` globale pour mieux contrôler l'accès aux AWS ressources en fonction de la valeur de l'identité de la source dans les demandes suivantes.

La politique de confiance de rôle suivante autorise l'utilisateur IAM `AdminUser`à endosser un rôle dans le compte `111122223333`. Elle octroie également l'autorisation à l'interface `AdminUser` de définir une identité source, dans la mesure où l'identité source définie correspond à `DiegoRamirez`.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAdminUserAssumeRole",
            "Effect": "Allow",
            "Principal": {"AWS": " arn:aws:iam::111122223333:user/AdminUser"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
                "StringEquals": {"sts:SourceIdentity": "DiegoRamirez"}
            }
        }
    ]
}
```

Pour en savoir plus sur les informations relatives à l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.

ensembles : TaskPolicyArn

Fonctionne avec des opérateurs ARN.

Utilisez cette clé pour comparer l'ARN de politique dans une AssumeRoot requête sts : avec l'ARN de politique spécifié dans la politique.

Disponibilité : cette clé est présente dans la demande lorsque vous faites une demande à l'aide de sts : AssumeRoot.

Les administrateurs peuvent utiliser cette clé de condition dans les politiques IAM pour empêcher certains rôles ou utilisateurs du compte de gestion ou du compte d'administrateur délégué d'effectuer certaines actions lorsqu'ils utilisent les informations d'identification racine. Pour de plus amples informations, veuillez consulter Effectuez une tâche privilégiée sur un AWS Organizations compte membre.

ensembles : TransitiveTagKeys

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour comparer les clés des balises de session transitoire de la demande avec celles spécifiées dans la politique.

Availability (Disponibilité) : cette clé est présente dans la demande lorsque vous effectuez une demande à l'aide d'informations d'identification de sécurité temporaires. Il s'agit notamment des informations d'identification créées à l'aide d'une opération assume-rôle, ou de l'opération `GetFederationToken`.

Lorsque vous effectuez une demande à l'aide d'informations d'identification de sécurité temporaires, le contexte de la demande inclut la clé de contexte `aws:PrincipalTag`. Cette clé inclut une liste de balises de session, de balises de session transitoire et de balises de rôle. Les balises de session transitoires sont des balises qui perdurent pendant toutes les sessions ultérieures lorsque vous utilisez les informations d'identification de session pour endosser un autre rôle. Le fait d'endosser un rôle à partir d'un autre est appelé chaînage des rôles.

Vous pouvez utiliser cette clé de condition dans une politique pour exiger la définition de balises de session spécifiques comme transitoires lorsque vous endossez un rôle ou fédérez un utilisateur.

Actions, ressources et clés de condition pour les AWS services

Chaque AWS service peut définir des actions, des ressources et des clés de contexte de condition à utiliser dans les politiques IAM. Pour obtenir la liste des AWS services et de leurs actions, ressources et clés contextuelles de condition, consultez la section Actions, ressources et clés de condition dans la référence d'autorisation de service.

Ressources pour en savoir plus sur IAM

L'IAM est un produit riche, et vous trouverez de nombreuses ressources pour vous aider à en savoir plus sur la manière dont l'IAM peut vous aider à sécuriser vos Compte AWS ressources.

Rubriques

- Identités
- Informations d'identification (mots de passe, clés d'accès et dispositifs MFA)
- Autorisations et politiques
- Fédération et délégation
- IAM et autres AWS produits
- Pratiques de sécurité générales
- Ressources générales

Identités

Consultez les ressources suivantes pour en savoir plus sur la création, la gestion et l'utilisation d'identités.

- Gestion des identités dans IAM Identity Center : informations procédurales relatives à la création d'utilisateurs et de groupes dans IAM Identity Center.
- IAM identités: discussion approfondie sur les utilisateurs, les groupes et les rôles.

Informations d'identification (mots de passe, clés d'accès et dispositifs MFA)

Consultez les guides suivants pour gérer les mots de passe, les clés d'accès et les appareils MFA pour vous Compte AWS et pour les utilisateurs IAM.

- Mots de passe utilisateur dans AWS : décrit les options disponibles pour la gestion des mots de passe des utilisateurs IAM de votre compte.
- Gestion des clés d'accès pour les utilisateurs IAM : décrit le fonctionnement des clés d'accès et la manière dont vous pouvez les utiliser pour effectuer des appels programmatiques à AWS. Il existe d'autres alternatives plus sécurisées aux clés d'accès que nous vous recommandons d'envisager

d'abord. Pour plus d'informations, consultez la section Considérations et alternatives relatives aux clés d'accès à long terme dans le Guide de Références générales AWS .

- AWS Multi-factor authentification dans IAM : décrit comment configurer votre compte et les utilisateurs IAM de manière à exiger à la fois un mot de passe et un code à usage unique qui est généré sur un appareil avant que la connexion soit autorisée. (Ceci est parfois appelé authentification à deux facteurs.)

Pour obtenir des informations générales sur les types d'informations d'identification que vous utilisez pour accéder à Amazon Web Services, consultez Informations d'identification de sécuritéAWS dans le Guide de Références générales AWS .

Autorisations et politiques

Découvrez le fonctionnement interne des politiques IAM, ainsi que les meilleures façons d'octroyer des autorisations :

- Politiques et autorisations dans Gestion des identités et des accès AWS : présente le langage de politique utilisé pour définir les autorisations. Décrit comment les autorisations peuvent être attachées à des utilisateurs ou des groupes ou, dans le cas de certains produits AWS , aux ressources proprement dites.
- Référence de l'élément de politique JSON IAM : fournit des descriptions et des exemples de chaque élément de langage de politique.
- Validation de politique IAM – recherche de ressources pour la validation de la politique JSON.
- Exemples de politiques basées sur l'identité IAM— Affiche des exemples de politiques pour les tâches courantes dans différents AWS produits.
- Générateur de politiques AWS : permet de créer des politiques personnalisées en choisissant des produits et des actions dans une liste.
- Simulateur de politiques IAM : testez si une politique autoriserait ou refuserait une demande spécifique à AWS.

Fédération et délégation

Vous pouvez accorder l'accès aux ressources de votre Compte AWS compte aux utilisateurs authentifiés (connectés) par un autre intermédiaire. Il peut s'agir d'utilisateurs IAM appartenant à un autre fournisseur Compte AWS (appelé délégation), d'utilisateurs authentifiés par le processus de

connexion de votre organisation ou d'utilisateurs d'un fournisseur d'identité Internet tel que Login with Amazon, Facebook, Google ou tout autre fournisseur d'identité compatible OpenID Connect (OIDC). Dans ces cas, les utilisateurs obtiennent des informations d'identification de sécurité temporaires pour accéder aux AWS ressources.

- Tutoriel IAM : accès délégué via AWS comptes utilisant des rôles IAM : vous guide pour octroyer un accès intercompte à un utilisateur IAM dans un autre Compte AWS.
- Scénarios courants d'informations d'identification temporaires— Décrit les moyens par lesquels les utilisateurs peuvent être fédérés AWS après s'être authentifiés en dehors de. AWS

IAM et autres AWS produits

La plupart AWS des produits sont intégrés à IAM afin que vous puissiez utiliser les fonctionnalités IAM pour protéger l'accès aux ressources de ces produits. Les ressources suivantes traitent de l'IAM et de la sécurité de certains des AWS produits les plus populaires. Pour une liste complète des produits fonctionnant avec IAM, avec des liens vers des informations complémentaires sur chacun de ces produits, consultez AWS services qui fonctionnent avec IAM.

Utilisation d'IAM avec Amazon EC2

- Contrôle de l'accès aux ressources Amazon EC2 : décrit comment utiliser les fonctions IAM pour permettre aux utilisateurs de gérer des instances Amazon EC2, des volumes, et plus encore.
- Utilisation des profils d'instance— Décrit comment utiliser les rôles IAM pour fournir en toute sécurité des informations d'identification pour les applications qui s'exécutent sur des instances Amazon EC2 et qui ont besoin d'accéder à d'autres AWS produits.

Utilisation d'IAM avec Amazon S3

- Gestion des autorisations d'accès à vos ressources Amazon S3 : présente le modèle de sécurité Amazon S3 pour les compartiments et les objets, notamment les politiques IAM.
- Rédaction de politiques IAM : accorder l'accès aux User-Specific dossiers d'un compartiment Amazon S3 — Explique comment permettre aux utilisateurs de protéger leurs propres dossiers dans Amazon S3. (Pour plus de billets se rapportant à Amazon S3 et IAM, sélectionnez la balise S3 sous le titre du billet de blog.)

Utilisation d'IAM avec Amazon RDS

- Utilisation de Gestion des identités et des accès AWS (IAM) pour gérer l'accès aux ressources Amazon RDS — Décrit comment utiliser IAM pour contrôler l'accès aux instances de base de données, aux instantanés de base de données, etc.
- Présentation des Resource-Level autorisations RDS — Décrit comment utiliser IAM pour contrôler l'accès à des instances Amazon RDS spécifiques.

Utilisation d'IAM avec Amazon DynamoDB

- Utilisation d'IAM pour contrôler l'accès aux ressources DynamoDB : décrit comment utiliser IAM pour permettre aux utilisateurs de gérer les tables et les index DynamoDB.
- La vidéo suivante (08:55) explique comment fournir un contrôle d'accès pour des éléments de base de données ou des attributs DynamoDB individuels (ou les deux).

  Démarrez avec le contrôle Fine-Grained d'accès pour DynamoDB

Pratiques de sécurité générales

Trouvez des conseils et des conseils d'experts sur les meilleurs moyens de sécuriser vos ressources Compte AWS et ressources :

- Meilleures pratiques en matière de sécurité, d'identité et de conformité — Trouvez des ressources sur la façon de gérer la sécurité de tous Comptes AWS les produits, y compris des suggestions concernant l'architecture de sécurité, l'utilisation de l'IAM, le chiffrement et la sécurité des données, etc.
- Gestion des identités et des accès : le AWS Well-Architected Framework vous aide à comprendre les concepts clés, les principes de conception et les meilleures pratiques architecturales pour la conception et l'exécution de charges de travail dans le cloud.
- Bonnes pratiques de sécurité dans IAM : fournit des recommandations sur les différentes façons d'utiliser IAM pour sécuriser votre Compte AWS et vos ressources.
- AWS CloudTrail Guide de l'utilisateur : AWS CloudTrail permet de suivre l'historique des appels d'API effectués AWS et de stocker ces informations dans des fichiers journaux. Vous pouvez ainsi identifier les utilisateurs et les comptes ayant accédé aux ressources de votre compte lors des appels, déterminer les actions demandées, etc.

Ressources générales

Explorez les ressources suivantes pour en savoir plus sur IAM et AWS.

- Informations sur le produit IAM : informations générales sur le produit Gestion des identités et des accès AWS .
- AWS re:Post pour Gestion des identités et des accès AWS — Visitez AWS re:Post pour discuter de questions techniques liées à l'IAM avec la AWS communauté.

- Cours et ateliers — Liens vers des cours axés sur les rôles et des cours spécialisés, ainsi que des ateliers à votre rythme pour vous aider à affiner vos AWS compétences et à acquérir une expérience pratique.
- AWS Centre pour développeurs : découvrez des didacticiels, téléchargez des outils et découvrez les événements AWS destinés aux développeurs.
- AWS Outils pour développeurs : liens vers des outils de développement, des kits de développement, des boîtes à outils IDE et des outils de ligne de commande pour développer et gérer des AWS applications.
- Centre de ressources pour la mise en route : découvrez comment configurer votre application Compte AWS, rejoindre la AWS communauté et lancer votre première application.
- Hands-On Tutoriels — Suivez les didacticiels étape par étape pour lancer votre première application le AWS.
- AWS Livres blancs : liens vers une liste complète de AWS livres blancs techniques, couvrant des sujets tels que l'architecture, la sécurité et l'économie et rédigés par AWS des architectes de solutions ou d'autres experts techniques.
- AWS Support Center — La plateforme de création et de gestion de vos AWS Support dossiers. Inclut également des liens vers d'autres ressources utiles, telles que des forums, des FAQ techniques, l'état de santé du service et AWS Trusted Advisor.
- Support— La page Web principale contenant des informations sur Support, un canal d'assistance personnalisé et rapide pour vous aider à créer et à exécuter des applications dans le cloud.
- Contactez-nous : point de contact central pour toute question relative à la facturation AWS , à votre compte, aux événements, à des abus ou à d'autres problèmes.
- AWS Conditions du site : informations détaillées sur nos droits d'auteur et notre marque déposée, votre compte, votre licence et l'accès au site, ainsi que sur d'autres sujets.

Appel de l'API IAM à l'aide de requêtes HTTP

Table des matières

- Points de terminaison
- HTTPS requis
- Signature des demandes d'API IAM

Vous pouvez accéder à l'IAM et aux AWS STS services par programmation à l'aide de l'API Query. Les demandes d'API de requête sont des demandes HTTPS qui doivent comporter un paramètre `Action` qui indique l'action à effectuer. IAM et AWS STS support des requêtes GET et POST pour toutes les actions. Autrement dit, l'API ne requiert pas l'utilisation de GET pour certaines actions et de POST pour d'autres. Toutefois, les requêtes GET sont soumises aux restrictions de taille applicables à une URL ; si cette limite varie en fonction du navigateur, elle est généralement de 2 048 octets. Par conséquent, dans le cas de demandes d'API de requête requérant des tailles plus importantes, il convient d'utiliser une requête POST.

Vous obtenez une réponse sous la forme d'un document XML. Pour plus d'informations sur la réponse, veuillez consulter les pages des actions spécifiques dans la Référence d'API IAM ou la Référence d'API AWS Security Token Service.

> **Tip**
>
> Au lieu de passer des appels directs aux opérations IAM ou AWS STS API, vous pouvez utiliser l'un des AWS kits de développement logiciel. Les AWS SDK se composent de bibliothèques et d'exemples de code pour divers langages de programmation et plates-formes (Java, Ruby, .NET, iOS, Android, etc.). Les kits SDK constituent un moyen pratique de créer un accès programmatique à IAM et. AWS Par exemple, ils automatisent les tâches telles que la signature cryptographique des demandes (voir ci-dessous), la gestion des erreurs et les nouvelles tentatives de demande. Pour plus d'informations sur les AWS kits SDK, notamment sur la manière de les télécharger et de les installer, consultez la page Outils pour Amazon Web Services.

Pour obtenir des détails sur les actions d'API et les erreurs, veuillez consulter la Référence des API IAM ou la Référence des API AWS Security Token Service.

Points de terminaison

IAM et AWS STS chacun d'entre eux disposent d'un seul point de terminaison mondial :

- (AIM) https://iam.amazonaws.com
- (AWS STS) https://sts.amazonaws.com

> **Important**
>
> AWS STS prend également en charge l'envoi de demandes à des points de terminaison régionaux en plus du point de terminaison global. AWS recommande d'utiliser des points de terminaison régionaux plutôt que des points de terminaison mondiaux pour réduire la latence, intégrer la redondance et augmenter la validité des jetons de session. Avant de pouvoir utiliser AWS STS dans une région, vous devez d'abord activer STS dans cette région pour votre Compte AWS. Pour plus d'informations sur l'activation de régions supplémentaires pour AWS STS, consultezGérer AWS STS dans un Région AWS.

Pour plus d'informations sur les AWS points de terminaison et les régions pour tous les services, voir Points de terminaison de service et quotas dans le. Références générales AWS

HTTPS requis

Dans la mesure où l'API de requête retourne des informations sensibles telles que les informations d'identification de sécurité, vous devez utiliser HTTPS avec toutes les demandes d'API.

Signature des demandes d'API IAM

Les demandes doivent être signées à l'aide d'un identifiant de la clé d'accès et d'une clé d'accès secrète. Nous vous recommandons vivement de ne pas utiliser vos Utilisateur racine d'un compte AWS informations d'identification dans le cadre de vos activités quotidiennes avec IAM. Vous pouvez utiliser les informations d'identification d'un utilisateur IAM ou vous pouvez les utiliser AWS STS pour générer des informations d'identification de sécurité temporaires.

Pour signer vos demandes d'API, nous vous recommandons d'utiliser AWS Signature Version 4. Pour plus d'informations sur l'utilisation de Signature Version 4, consultez Processus de signature Signature Version 4 dans la Référence générale d'AWS .

Si vous devez utiliser Signature Version 2, des informations sur son utilisation sont disponibles dans le document Référence générale AWS.

Pour plus d'informations, veuillez consulter les ressources suivantes :

- AWS Informations d'identification de sécurité. Fournit des informations générales sur les types d'informations d'identification utilisés pour accéder AWS.
- Bonnes pratiques de sécurité dans IAM. Présente une liste de suggestions relatives à l'utilisation du service IAM afin de sécuriser vos AWS ressources.
- Informations d'identification de sécurité temporaires dans IAM. Décrit comment créer et utiliser les informations d'identification de sécurité temporaires.

Historique du document pour IAM

Le tableau suivant décrit les principales mises à jour de la documentation pour IAM.

Modification	Description	Date
job_workflow_ref accepté comme revendication de portée alternative pour le fournisseur Actions OIDC GitHub	Pour le fournisseur GitHub Actions OIDC, IAM accepte désormais la `sub` réclamation ou la réclamation en tant que contrôle du `job_workflow_ref` fournisseur d'identité dans les politiques de confiance des rôles.	12 mai 2026
Clés de contexte de condition IAM pour les API d’informations d’identification spécifiques au service	IAM prend désormais en charge deux nouvelles clés de contexte de condition pour contrôler l’accès aux API d’informations d’identification spécifiques aux services : `iam:ServiceSpecificCredentialAgeDays` et `iam:ServiceSpecificCredentialServiceName` . Ces clés vous permettent de restreindre la création et la gestion des informations d'identification spécifiques à un service en fonction des paramètres d'expiration et des services autorisés AWS . • `iam:ServiceSpecificCredentialAgeDays`	4 septembre 2025

	• iam:ServiceSpecificCredentialServiceName	
Clés de condition des points de terminaison d’un VPC pour les contrôles du périmètre réseau	IAM prend désormais en charge trois nouvelles clés contextuelles de condition AWS globales pour la mise en œuvre de contrôles périmétriques évolutifs du réseau : aws:VpceAccount aws:VpceOrgID , etaws:VpceOrgPaths . Ces clés permettent de garantir que les demandes transitent par les points de terminaison d’un VPC appartenant à des comptes, des organisations ou des unités organisationnelles spécifiques, et s’adaptent automatiquement à votre utilisation des points de terminaison d’un VPC sans nécessiter de mise à jour des politiques lorsque vous créez de nouveaux points de terminaison. Pour plus d’informations, consultez Mettre en place des barrières de protection des autorisations à l’aide de périmètres de sécurisation des données.	28 août 2025

Tutoriels sur la fédération IAM SAML CloudFormation	IAM a ajouté de nouveaux didacticiels pour créer des fournisseurs d'identité SAML (IdPs) et des rôles fédérés à l'aide de. CloudFormation • Créer un IdP SAML avec CloudFormation • Créez un rôle fédéré SAML avec CloudFormation • Créez un IdP SAML et un rôle fédéré avec CloudFormation	25 août 2025
Contrôles supplémentaires pour les fournisseurs OIDC partagés	IAM inclut désormais Amazon Cognito, Azure Sentinel, Pulumi Cloud et le point de terminaison global Vercel dans la liste des fournisseurs d’identité OIDC partagés qui nécessitent une évaluation explicite des demandes spécifiques dans les jetons Web JSON (JWT).	1er août 2025

L'analyseur d'accès IAM a ajouté des analyseurs d'accès internes	L'analyseur d'accès IAM vous aide à identifier les principaux de votre organisation ou de votre compte qui ont accès à certaines ressources critiques pour l'entreprise. Les analyseurs d'accès internes prennent en charge la mise en œuvre du principe du moindre privilège en garantissant que vos ressources spécifiées ne sont accessibles qu'aux principaux désignés au sein de votre organisation ou de votre compte AWS .	16 juin 2025
Contrôles des fournisseurs d'identité pour les fournisseurs OIDC partagés	IAM exige désormais une évaluation explicite des revendications spécifiques dans les jetons Web JSON (JWT) pour les fournisseurs d'identité OIDC partagés reconnus. Ce contrôle de sécurité garantit que seules les identités autorisées de l'organisation concernée peuvent assumer des rôles et accéder aux AWS ressources.	6 juin 2025
`IAMUserChangePassword` : autorisations ajoutées	IAM a ajouté des autorisations à IAMUserChangePassword pour autoriser les utilisateurs spécifiés dans un chemin.	28 mai 2025

AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté `iam:GetAccountAuthorizationDetails` sur les autorisations de niveau de service de AccessAnalyzerServiceRolePolicy.	12 mai 2025
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	IAM Access Analyzer a ajouté des points d'accès au compartiment de répertoires Amazon S3 aux autorisations de niveau de service de. AccessAnalyzerServiceRolePolicy	31 mars 2025
Prise en charge des points de terminaison double pile IAM	IAM offre désormais une prise en charge améliorée des points de terminaison double pile, ce qui permet aux clients de communiquer avec IAM à l'aide d'adresses IPv4 ou IPv6.	20 mars 2025
Mises à jour de la logique d'évaluation des politiques	Documentation améliorée pour la logique d'évaluation des politiques, y compris des organigrammes améliorés et des explications plus claires sur la manière dont AWS les politiques sont évaluées pour déterminer s'il faut autoriser ou refuser une demande.	30 janvier 2025
IAMDeleteRootUserCredentials – Supprimer des autorisations	IAM a supprimé l'autorisation `iam:DeleteVirtualMFADevice` de la politique gérée.	7 janvier 2025

AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté la prise en charge de l'autorisation de récupérer des informations sur les paramètres du compte Amazon ECR et les politiques de registre aux autorisations de niveau de service de AccessAnalyzerServiceRolePolicy.	10 décembre 2024
Ajout d'une configuration d'accès à l'analyseur d'accès IAM	L'analyseur d'accès IAM a ajouté la prise en charge de la configuration des analyseurs afin de modifier l'étendue dans laquelle les Comptes AWS, les utilisateurs IAM et les rôles génèrent des résultats.	14 novembre 2024

Gestion centralisée de l’accès root pour les comptes membres	Vous pouvez désormais gérer les informations d’identif ication des utilisateurs racine privilégiés sur l’ensemble des comptes membres d’ AWS Organizations grâce à un accès root centralisé. Sécurisez de manière centralis ée les informations d'identif ication de l'utilisateur root de votre utilisation Comptes AWS gérée AWS Organizations pour supprimer et empêcher la récupération des informati ons d'identification des utilisate urs root et l'accès à grande échelle. • Meilleures pratiques pour les utilisateurs root	14 novembre 2024

AWS mise à jour des politiques gérées - Nouvelles politiques	IAM a ajouté deux nouvelles politiques pour définir les autorisations pour les sessions d'utilisateur racine privilégiés que vous pouvez démarrer après avoir centralisé l'accès des utilisateurs racine pour les comptes membres de votre organisation. • IAMAuditRootUserCredentials • IAMCreateRootUserPassword • IAMDeleteRootUserCredentials • S3UnlockBucketPolicy • SQSUnlockQueuePolicy	14 novembre 2024
Prise en charge des politiques de contrôle des AWS Organizations ressources (RCP)	Utilisez une politique de contrôle AWS Organizations des ressources (RCP) pour définir les autorisations maximales pour les ressources au sein des comptes de votre organisation ou de votre unité organisationnelle (OU). Les RCP limitent les autorisations que les politiques basées sur l'identité ou sur les ressources peuvent accorder aux ressources des comptes au sein de votre organisation.	13 novembre 2024

AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté la prise en charge de l'autorisation de récupérer des informations sur les balises de rôle et d'utilisateur IAM aux autorisations de niveau de service de AccessAnalyzerServiceRolePolicy.	29 octobre 2024
Améliorations de prise en charge du chiffrement SAML	Documentation améliorée pour la prise en charge du chiffrement SAML dans les fournisseurs SAML IAM, y compris des conseils de dépannage améliorés et des précisions sur la compatibilité des services.	5 juin 2024
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté la prise en charge de l'autorisation de récupérer des informations sur les politiques de rôle et d'utilisateur IAM aux autorisations de niveau de service de AccessAnalyzerServiceRolePolicy.	30 mai 2024

Prise en charge du chiffrement pour les fournisseurs d'identité SAML	Les fournisseurs SAML IAM prenent désormais en charge les assertions chiffrées dans la réponse SAML de votre IdP externe. Pour comprendre comment fonctionne le chiffrement avec la fédération IAM SAML, consultez la section Utilisation de la SAML-based fédération pour l'accès aux API.	4 février 2024
`AccessAnalyzerServiceRolePolicy` : autorisations ajoutées	L'analyseur d'accès IAM a ajouté la prise en charge de l'accès public des instantanés Amazon EC2 aux autorisations de niveau de service de AccessAnalyzerServiceRolePolicy.	23 janvier 2024
`AccessAnalyzerServiceRolePolicy` : autorisations ajoutées	IAM Access Analyzer a ajouté des flux et des tables DynamoDB aux autorisations de niveau de service de. AccessAnalyzerServiceRolePolicy	11 janvier 2024
`AccessAnalyzerServiceRolePolicy` : autorisations ajoutées	IAM Access Analyzer a ajouté des compartiments d'annuaire Amazon S3 aux autorisations de niveau de service de. AccessAnalyzerServiceRolePolicy	1er décembre 2023

IAMAccessAnalyzerReadOnlyAccess : autorisations ajoutées	IAM Access Analyzer a ajouté des autorisations pour vous IAMAccessAnalyzerReadOnlyAccess permettre de vérifier si les mises à jour de vos politiques accordent un accès supplémentaire. L'analyseur d'accès IAM a besoin de cette autorisation pour effectuer des vérifications de politique sur vos politiques.	26 novembre 2023
L'analyseur d'accès IAM a ajouté des analyseurs d'accès non utilisés	L'analyseur d'accès IAM simplifie l'inspection des accès non utilisés pour vous guider vers le moindre privilège . L'analyseur d'accès IAM analyse en permanence vos comptes pour identifier les accès non utilisés et crée un tableau de bord centralisé contenant les résultats.	26 novembre 2023
L'analyseur d'accès IAM a ajouté des vérifications de politiques personnalisées	L'analyseur d'accès IAM fournit désormais des vérifications de politiques personnalisées pour valider que les politiques IAM sont conformes à vos normes de sécurité avant les déploiements.	26 novembre 2023

AccessAnalyzerServiceRolePolicy : autorisations ajoutées	L'analyseur d'accès IAM a ajouté des actions IAM aux autorisations de niveau du service d'AccessAnalyzerServiceRolePolicy pour prendre en charge les actions suivantes : • Lister les entités pour une politique • Générer les dernières informations consultées sur le service • Lister les informations clés d'accès	26 novembre 2023
Prise en charge des informations de la dernière action consultée de l'élaboration de politiques pour plus de 60 services et actions supplémentaires	IAM prend désormais en charge les informations de la dernière action consultée et génère des politiques contenant des informations au niveau de l'action pour plus de 60 services supplémentaires, ainsi qu'une liste des actions pour lesquelles les informations de la dernière action consultée sont disponibles.	1er novembre 2023
Prise en charge des informations de la dernière action consultée pour plus de 140 services	IAM fournit désormais les informations de la dernière action consultée pour plus de 140 services, ainsi qu'une liste des actions pour lesquelles les informations de la dernière action consultée sont disponibles.	14 septembre 2023

Prise en charge des dispositifs d'authentification multifactorielle (MFA) pour les utilisateurs root et IAM	Vous pouvez désormais ajouter jusqu'à huit dispositifs MFA par utilisateur, y compris des clés de sécurité FIDO, un mot de passe à usage unique basé sur le temps (TOTP) associé à des applications d'authentification virtuelle ou des jetons TOTP matériels.	16 novembre 2022
Prise en charge de nouveaux types de ressources IAM Access Analyzer	L'IAM Access Analyzer a ajouté la prise en charge les types de ressources suivants : • Instantanés de volumes Amazon EBS • Référentiels Amazon ECR • Système de fichiers Amazon EFS • Instantanés de base de données Amazon RDS • Instantanés du cluster de base de données Amazon RDS • Rubriques Amazon SNS	25 octobre 2022
Obsolète et mise à jour de U2F WebAuthn/FIDO	Suppression de la mention d'U2F en tant qu'option MFA et ajout d'informations sur les clés de sécurité WebAuthn FIDO2 et FIDO.	31 mai 2022

Mises à jour de la résilience dans IAM	Ajout d'informations sur le maintien de l'accès aux informations d'identification IAM lorsqu'un événement perturbe la communication entre les Régions AWS.	16 mai 2022
Nouvelles clés de condition globales pour les ressources	Vous pouvez désormais contrôler l'accès aux ressources en fonction du compte, de l'unité organisationnelle (UO) ou de l'organisation AWS Organizations qui contient vos ressources. Vous pouvez utiliser les clés de condition globales `aws:ResourceAccount`, `aws:ResourceOrgID` et `aws:ResourceOrgPaths` dans une politique IAM.	27 avril 2022
Exemples de code pour IAM à l'aide AWS de kits de développement logiciel	Ajout d'exemples de code qui montrent comment utiliser IAM avec un kit de développement AWS logiciel (SDK). Les exemples sont divisés en extraits de code qui vous montrent comment appeler des fonctions de service individuelles et en exemples qui vous montrent comment accomplir une tâche spécifique en appelant plusieurs fonctions au sein d'un même service.	7 avril 2022

Mise à jour de l'organigramme logique d'évaluation des politiques	Mises à jour de l'organigramme logique d'évaluation des politiques et du texte associé dans la section déterminer si une demande est autorisée ou rejetée dans un compte.	17 novembre 2021
Mises à jour des bonnes pratiques de sécurité	Ajout d’informations sur la création d’utilisateurs administratifs au lieu d’utiliser les informations d’identification de l’utilisateur racine, suppression des pratiques exemplaires consistant à utiliser des groupes IAM pour attribuer des autorisations aux utilisateurs IAM, et clarification des cas dans lesquels il convient d’utiliser des politiques gérées au lieu de politiques en ligne.	5 octobre 2021
Mises à jour de la rubrique logique d'évaluation des politiques pour les stratégies basées sur les ressources	Ajout d'informations sur l'impact des politiques basées sur les ressources et des différents types de principaux dans le même compte.	5 octobre 2021
Mises à jour des clés de condition à valeur unique et à valeurs multiples	Les différences entre les clés de condition à valeur unique et à valeurs multiples sont maintenant expliquées plus en détail. Le type de valeur a été ajouté à chaque clé de contexte de condition globale AWS.	30 septembre 2021

L'analyseur d'accès IAM prend en charge les points d'accès Amazon S3 Multi-Region	IAM Access Analyzer identifie les compartiments Amazon S3 qui autorisent l'accès public et intercomptes, y compris ceux qui utilisent des points d'accès Amazon S3. Multi-Region	2 septembre 2021
AWS mises à jour des politiques gérées : mise à jour d'une politique existante	IAM Access Analyzer a mis à jour une politique AWS gérée existante.	2 septembre 2021
Davantage de services pris en charge pour la génération de politiques au niveau action	IAM Access Analyzer peut générer des politiques IAM avec des informations d'activité d'accès au niveau de l'action pour des services supplémentaires. AWS	24 août 2021
Générer des politiques IAM pour des journaux d'activité entre comptes	Vous pouvez désormais utiliser IAM Access Analyzer pour générer des politiques précises en fonction de votre activité d'accès à l'aide d'un historique dans AWS CloudTrail un autre compte, par exemple un suivi centralisé. AWS Organizations	18 août 2021

Vérifications de politiques IAM Access Analyzer supplémentaires	IAM Access Analyzer a étendu la validation de politique en ajoutant de nouvelles vérifications de politique qui valident les conditions incluses dans les politiques IAM. Ces vérifications analysent le bloc de condition dans votre instruction de politique et signalent les avertissements de sécurité, les erreurs et les suggestions, ainsi que des recommandations exploitables. IAM Access Analyzer a ajouté les vérifications de politique suivantes : • Erreur : format de principal de service non valide • Erreur : clé de balise manquante dans la condition • Avertissement de sécurité — Refus NotAction avec clé de condition d'étiquette non prise en charge pour le service • Avertissement de sécurité : refuser avec une clé de condition de balise non prise en charge pour le service • Avertissement de sécurité : clés de condition appariées manquantes	29 juin 2021

	• Suggestion — Autoriser NotAction avec une clé de condition de balise non prise en charge pour le service • Suggestion : autoriser avec la clé de condition de balise non prise en charge pour le service	
Prise en charge de dernière action consultée, pour davantage de services	Vous pouvez désormais afficher des informations relatives à la dernière action consultée dans la console IAM, à propos de la dernière fois où un principal IAM a utilisé une action pour les services suivants : actions de gestion Amazon EC2, IAM, Lambda et Amazon S3. Vous pouvez également utiliser l' AWS API AWS CLI or pour récupérer un rapport de données. Elles vous permetten t d'identifier toute autorisat ion inutile. Vous pouvez ainsi peaufiner vos politiques IAM afin qu'elles respectent au plus près le principe du moindre privilège.	19 avril 2021

Surveiller et contrôler les mesures prises avec les rôles endossés	Les administrateurs peuvent configurer les rôles IAM pour exiger des identités qu'elles transmettent une identité source, qui est journalisée dans AWS CloudTrail. L'examen des informations d'identité source aide les administrateurs à déterminer qui ou ce qui a effectué des actions avec les sessions de rôle endossées.	13 avril 2021
Générer des politiques IAM basées sur l'activité d'accès	Vous pouvez désormais utiliser IAM Access Analyzer pour générer des politiques affinées en fonction de l'activité d'accès trouvée dans votre AWS CloudTrail.	7 avril 2021
Vérifications de politiques IAM Access Analyzer	IAM Access Analyzer fournit désormais plus de 100 vérifications de politiques avec des recommandations exploitables lors de la création de politiques.	16 mars 2021
Options de validation de politique étendues	Validation étendue des politiques disponible dans la console IAM, l' AWS API et à AWS CLI l'aide de vérifications des politiques dans IAM Access Analyzer pour vous aider à créer des politiques JSON sécurisées et fonctionnelles.	15 mars 2021

Balisage des ressources IAM	Vous pouvez maintenant baliser des ressources IAM supplémentaires à l'aide d'une paire clé-valeur de balise.	11 février 2021
Politique de mot de passe par défaut des utilisateurs IAM	Si vous ne définissez pas de politique de mot de passe personnalisée pour votre Compte AWS, les mots de passe utilisateur IAM doivent désormais respecter la politique de AWS mot de passe par défaut.	18 novembre 2020
Les pages d'actions, de ressources et de clés de condition pour les AWS services ont été déplacées	Chaque AWS service peut définir des actions, des ressources et des clés contextuelles de condition à utiliser dans les politiques IAM. Vous pouvez désormais trouver la liste des AWS services et leurs actions, ressources et clés contextue lles de condition dans la référence d'autorisation des services.	16 novembre 2020

Durée de session de rôle plus longue des utilisateurs IAM	Les utilisateurs IAM peuvent désormais bénéficier d'une durée de session de rôle plus longue lorsqu'ils changent de rôle dans le Console de gestion AWS, ce qui réduit les interruptions dues à l'expirat ion de la session. Les utilisate urs se voient accorder la durée maximale de session définie pour le rôle, ou la durée restante de la session de l'utilisateur IAM, selon la valeur la moins élevée.	24 juillet 2020
Utilisez Service Quotas pour demander des augmentations rapides pour les entités IAM	Vous pouvez demander des augmentations de quota pour des quotas IAM ajustable s à l'aide de la console Service Quotas. Maintenant, certaines augmentations sont automatiquement approuvée s dans Service Quotas et disponibles dans votre compte en quelques minutes. Les demandes plus important es sont soumises à AWS Support.	25 juin 2020

Les dernières informations consultées dans IAM incluent maintenant les actions de gestion Amazon S3	Outre les données de dernier accès au service, vous pouvez désormais afficher des informations dans la console IAM concernant la dernière fois qu'un principal IAM a utilisé une action Amazon S3. Vous pouvez également utiliser l' AWS API AWS CLI or pour récupérer le rapport de données. Le rapport contient des informations sur les services autorisés et les actions auxquels les principaux ont tenté d'accéder pour la dernière fois et quand. Elles vous permettent d'identifier toute autorisation inutile. Vous pouvez ainsi peaufiner vos politiques IAM afin qu'elles respectent au plus près le principe du moindre privilège.	3 juin 2020
Ajout du chapitre sur la sécurité	Le chapitre sur la sécurité vous aide à comprendre comment configurer IAM et AWS STS atteindre vos objectifs de sécurité et de conformité. Vous apprendrez également à utiliser d'autres services AWS pour surveiller et sécuriser vos ressources IAM.	29 avril 2020

ensembles : RoleSessionName	Vous pouvez désormais écrire une politique qui accorde des autorisations en fonction du nom de session spécifié par un principal lors de l'attribution d'un rôle.	21 avril 2020
AWS mise à jour de la page de connexion	Lorsque vous vous connectez sur la page de AWS connexion principale, vous pouvez désormais choisir de vous connecter en tant qu'utilisateur IAM Utilisateur racine d'un compte AWS ou en tant qu'utilisateur. Lorsque vous le faites, l'étiquette sur la page indique si vous devez fournir votre adresse e-mail d'utilisateur root ou les informations de votre utilisateur IAM. Cette documentation inclut des captures d'écran mises à jour pour vous aider à comprendre les pages de connexion AWS .	4 mars 2020

AWS : via AWS Service et aws : clés de condition CalledVia	Vous pouvez désormais écrire une politique pour limiter le fait que les services puissent effectuer des demandes au nom d'un principal IAM (utilisateur ou rôle). Lorsqu'un principal effectue une demande à un service AWS , ce service peut utiliser les informations d'identification du principal pour effectuer des demandes ultérieures à d'autres services. Utilisez la clé de condition `aws:ViaAWSService` pour déterminer si un service effectue une demande à l'aide des informations d'identif ication d'un principal. Utilisez la clé de condition `aws:CalledVia` pour déterminer si des services spécifiques effectuen t une demande à l'aide des informations d'identification d'un principal.	20 février 2020
Ajout de la prise en charge des limites d'autorisations par le simulateur de politiques	Vous pouvez désormais tester l'effet des limites d'autoris ations sur les entités IAM à l'aide du simulateur de politique IAM.	23 janvier 2020

Cross-account évaluation des politiques	Vous pouvez désormais découvrir comment AWS évalue les politiques d'accès entre comptes. Cette opération se produit lorsqu'une ressource d'un compte d'approbation inclut une politique basée sur la ressource permettant à un principal d'un autre compte d'accéder à la ressource. La demande doit être autorisée dans les deux comptes.	2 janvier 2020
Balises de session	Vous pouvez désormais inclure des balises lorsque vous endossez un rôle ou fédérez un utilisateur dans AWS STS. Lorsque vous effectuez les opérations `GetFederationToken` ou `AssumeRole` , vous pouvez transmettre les balises de session en tant qu'attrib uts. Lorsque vous effectuez les `AssumeRoleWithWebI dentity` opérations `AssumeRoleWithSAML` ou, vous pouvez transmettre des attributs de vos identités d'entreprise à AWS.	22 novembre 2019

Contrôlez l'accès pour les groupes Comptes AWS d'utilisateurs AWS Organizations	Vous pouvez désormais référencer des unités organisationnelles (UO) depuis AWS Organizations les politiques IAM. Si vous avez l' AWS Organizations habitude d'organiser vos comptes en unités d'organisation, vous pouvez exiger que les principaux appartiennent à une unité d'organisation spécifique avant d'accorder l'accès à vos ressources. Les principaux incluent Utilisateur racine d'un compte AWS les utilisateurs IAM et les rôles IAM. Pour ce faire, indiquez le chemin d'unité d'organisation dans la clé de condition `aws:PrincipalOrgPaths` de vos politiques.	20 novembre 2019
Dernier rôle utilisé	Vous pouvez désormais afficher la date, l'heure et la région auxquelles un rôle a été utilisé pour la dernière fois. Ces informations vous aident également à identifier les rôles inutilisés dans votre compte. Vous pouvez utiliser l' Console de gestion AWS AWS API AWS CLI et pour afficher des informations sur la date de dernière utilisation d'un rôle.	19 novembre 2019

Mise à jour de la page des clés de contexte de condition globale	Vous pouvez désormais savoir quand chacune des clés de condition globale est incluse dans le contexte d'une requête. Vous pouvez également accéder à chaque clé plus facilement à l'aide de la table des matières de la page. Les informations sur la page vous aident à rédiger des politiques plus précises. Par exemple, si vos employés utilisent la fédération avec des rôles IAM, vous devez utiliser la clé `aws:userId` et non la clé `aws:userName` . La clé `aws:userName` s'applique uniquement aux utilisateurs IAM et non aux rôles.	6 octobre 2019

ABAC dans AWS	Découvrez comment le contrôle d'accès basé sur les attributs (ABAC) AWS utilise des balises et comment il se compare au modèle d'autorisation traditionnel AWS . Utilisez le didacticiel ABAC pour apprendre à créer et tester une politique qui permet aux rôles IAM avec des balises principales d'accéder aux ressources avec des balises correspondantes. Cette stratégie permet aux utilisateurs de consulter ou de modifier uniquement les AWS ressources nécessaires à leur travail.	3 octobre 2019
AWS STS GetAccessKeyInfo fonctionnement	Vous pouvez consulter les clés AWS d'accès contenues dans votre code pour déterminer si elles proviennent d'un compte que vous possédez. Vous pouvez transmettre un identifiant de clé d'accès à l'aide de la `aws sts get-access-key-info` AWS CLI commande ou de l'opération `GetAccessKeyInfo` AWS API.	24 juillet 2019

Affichage des dernières informations du AWS Organizations service consultées dans IAM	Vous pouvez désormais consulter les informations relatives au dernier accès au service pour une AWS Organizations entité ou une politique dans la AWS Organizations section de la console IAM. Vous pouvez également utiliser l' AWS API AWS CLI or pour récupérer le rapport de données. Ces données incluent des informations sur les services autorisés auxquels les administrateurs d'un AWS Organizations compte ont tenté d'accéder pour la dernière fois et à quelle date. Vous pouvez utiliser ces informations pour identifier les autorisations inutiles afin d'affiner vos AWS Organizations politiques afin de mieux respecter le principe du moindre privilège.	20 juin 2019
Utilisation d'une politique gérée en tant que politique de session	Vous pouvez désormais transmettre jusqu'à 10 ARN de politique gérée lorsque vous endossez un rôle. Ceci vous permet de limiter les autorisations des informations d'identification temporaires du rôle.	7 mai 2019

AWS STS Compatibilité régionale des jetons de session pour le point de terminaison global	Vous pouvez désormais choisir d'utiliser la version 1 ou la version 2 des jetons du point de terminaison global. Les jetons de version 1 ne sont valides que dans les AWS régions disponibles par défaut. Ces jetons ne fonctionnent pas dans les régions activées manuellement, par exemple Asie-Pacifique (Hong Kong). Les jetons de la version 2 sont valides dans toutes les régions. Toutefois, les jetons de la version 2 sont plus longs et peuvent avoir un impact sur les systèmes où vous stockez temporairement les jetons.	26 avril 2019
Autoriser l'activation et la désactivation des régions AWS	Vous pouvez désormais créer une politique qui autorise un administrateur à activer et désactiver la région Asie-Pacifique (Hong Kong) (ap us-east-1).	24 avril 2019

Page My Security Credentials (Mes informations d'identification de sécurité) de l'utilisateur IAM	Les utilisateurs IAM peuvent désormais gérer leurs propres informations d'identification sur la page My Security Credentials (Mes informations d'identification de sécurité). Cette Console de gestion AWS page affiche les informations relatives au compte, telles que l'ID du compte et l'ID utilisateur canonique. Les utilisateurs peuvent également consulter et modifier leurs propres mots de passe, clés d'accès, X.509 certificats, clés SSH et informations d'identification Git.	24 janvier 2019
API de l’analyseur d’accès	Vous pouvez désormais utiliser l' AWS API AWS CLI et pour afficher les informations relatives aux derniers accès au service.	7 décembre 2018

Balisage des utilisateurs et des rôles IAM	Vous pouvez désormais utiliser les balises IAM pour ajouter des attributs personnalisés à une identité (utilisateur ou rôle IAM) à l'aide d'une paire de balises clé-valeur. Vous pouvez également utiliser des balises pour contrôler l'accès aux ressources d'une identité ou pour contrôler les balises qui peuvent être attachées à une identité.	14 novembre 2018
Clés de sécurité U2F	Vous pouvez désormais utiliser les clés de sécurité U2F comme option d'authentification multi-facteurs (MFA, Multi-Factor Authentication) lors de la connexion à la Console de gestion AWS.	25 septembre 2018
Prise en charge des points de terminaison Amazon VPC	Vous pouvez désormais établir une connexion privée entre votre VPC et la région AWS STS de l'Ouest des États-Unis (Oregon).	31 juillet 2018
Limites d'autorisations	La nouvelle fonctionnalité facilite les employés approuvés à accorder la possibilité de gérer les autorisations IAM sans également accorder un accès administratif IAM complet.	12 juillet 2018

aws : PrincipalOrg ID	La nouvelle clé de condition permet de contrôler plus facilement l'accès aux AWS ressources en spécifian t l' AWS organisation des principaux IAM.	17 mai 2018
aws : RequestedRegion	La nouvelle clé de condition permet d'utiliser plus facilemen t les politiques IAM pour contrôler l'accès aux AWS régions.	25 avril 2018
Augmentation de la durée de la session pour les rôles IAM	Un rôle IAM peut désormais avoir une durée de session de 12 heures.	28 mars 2018
Mise à jour du flux de travail pour la création de rôles	Un nouveau flux de travail améliore le processus de création des relations d'approbation et d'attachement des autorisations aux rôles.	8 septembre 2017
Compte AWS processus de connexion	L'expérience de AWS connexion mise à jour permet à la fois à l'utilisateur root et aux utilisateurs IAM d'utiliser le lien Se connecter à la console sur la page d'accueil Console de gestion AWS de l'applica tion.	25 août 2017
Exemple de politiques IAM	La mise à jour de la documentation comprend plus de 30 exemples de politiques.	2 août 2017

Bonnes pratiques IAM	Les informations ajoutées dans la section Utilisateurs de la console IAM facilitent le respect des bonnes pratiques IAM.	5 juillet 2017
Ressources Auto Scaling	Resource-level les autorisations peuvent contrôler l'accès et les autorisations relatives aux ressources Auto Scaling.	16 mai 2017
Bases de données Amazon RDS for MySQL et Amazon Aurora	Les administrateurs de base de données peuvent associer des utilisateurs de base de données à des utilisateurs et à des rôles IAM et gérer ainsi l'accès des utilisateurs à toutes les AWS ressources à partir d'un emplacement unique.	24 avril 2017
Service-linked rôles	Service-linked les rôles constituent un moyen plus simple et plus sûr de déléguer des autorisations aux AWS services.	19 avril 2017
Récapitulatifs de la politique	De nouveaux récapitulatifs de la politique permettent de comprendre plus simplement les autorisations dans les politiques IAM.	23 mars 2017

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.